PRIJS ƒ 7.25/Bfr. 135

Commodore INFO

Listingtips

Basicode 3

GEOS nieuws

ONAFHANKELIJK BLAD VOOR COMMODORE GEBRUIKERS

JAARGANG 3, NO. 9, DEC. 1986



LISTINGS

Basic-uitbreiding

Commodore in 1987

C-16 tekens creëren

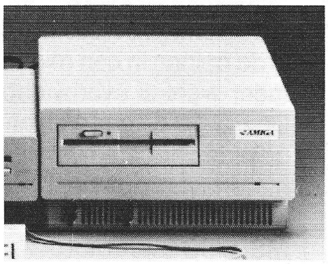
Leerprogramma's maken

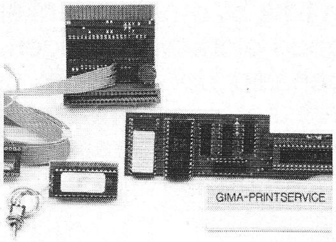
Hoge resolutie op 80-koloms scherm

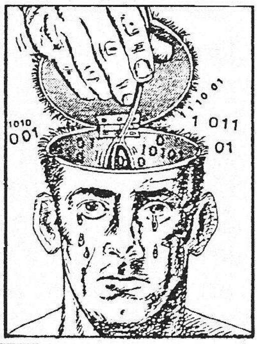
Vaste rubrieken:

- Machinetaal
- Zo zit dat
- Vragen
- Basic cursus
- Softwarebesprekingen

Inhoud van dit nummer









telefoon het telefoonnummer voor vragen over gepubliceerde listings gewijzigd. Alleen op maandag van 5 tot 9 uur 's avonds kan men bellen met 02155-25162. Het oude telefoonnummer wordt NIET meer bemand.



COMMODORE-INFO

Jaarg. 3, no. 9, dec. 1986

Uitgave: Sala Communications

Uitgever: Drs. J. Taverne

REDAKTIE

Ir. L. Sala hoofdredacteur
J. Bodzinga adj. hoofdred.
K. van der Vlies reportage
R. van den Heuvel listings
B. Munniksma onderwijs
Sj. Bakker machinetaal
U. Schuurmans software
R. Goudriaan listings

ART DEPARTMENT

Strip: Bert Tier

Illustraties: Ben van Mierlo, Ymmot

Advertentie-exploitatie: Ing. V. Sala, R. Akker

Redaktieadres: Postbus 112, 1260 AC Blaricum, ☎ 02152-65695

Abonnementen en administratie: Postbus 5570, 1007 AN Amsterdam, Den Texstraat 5a, 1017 XW Amsterdam, ☎ 020-273198

Vragen betreffende abonnementen bij voorkeur schriftelijk, met meesturen van het omslagetiket. Telefonisch uitsluitend van 10.00 tot 15.00 uur.

Abonnement: (10 nummers) *f* 55,– of Bfr. 1100 per jaar

Betaling op Giro 1585491 tnv. SAC/COMMODORE-INFO Blaricum of in België op Bank BBL nr. 310050602562, vermeld SAC/ COMMODORE-INFO. Oude nummers à *f* 6,75 alleen bij vooruitbetaling op een van bovenstaande rekeningen. Ook telefonische opgave voor een abonnement is mogelijk. Bel GRATIS 06-0224222, HP-Teleservice, elke dag tot 20.30 uur (ook in het weekend). Dit nummer is alleen voor telefonische opgave van NIEUWE abonnementen.

Redactiesecretariaat: Ron van Zalinge

Druk: Verweij, Mijdrecht, NDB, Zoeterwoude

Distributie:
in Nederland Betapress/Gilze
in België AMP/Brussel

ISSN: 0169-3085

Programmeerwestrijd 1986

Prijswinnaars

Voor onze grote programmeerwedstrijd hebben we een vele inzendingen ontvangen, waar een groot aantal leuke programma's bijzat. De deelname van de dames viel een beetje tegen, hoewel er toch een lichte stijging ten opzichte van vorig jaar te constateren valt.
Na vele uren zoeken, bekijken, lezen, testen en beoordelen zijn we er uiteindelijk toch uitgekomen. Als Uw programma door ons niet uitgekozen is als prijswinnend betekent dit niet, dat het onvoldoende van kwaliteit is maar we hebben een keuze moeten maken, waarbij we o.a. hebben beoordeeld op orginaliteit, onderwerp, lengte en hoe geprogrammeerd. Is Uw programma er door ons nu niet uitgekozen, dan kunt U toch nog winnen, want uit het materiaal dat er ingezonden is plaatsen we nog een aantal listings en in dat geval krijgt U daarvoor ook een vergoeding uitgekeerd.

De **hoofdprijs, een kleurenmonitor**, is gewonnen door Durk Bouma uit Nijland met een programma waar men flink bij moet nadenken als men het tot een goed einde wil brengen.
De tweede prijs is behaald door Marcel de Groot uit Hillegom met het programma diskhulp, dat voor vele van ons erg interessant is en veel tijd en moeite kan besparen. Hij krijgt hiervoor een diskdrive.
De derde prijs is voor Harrie van Lammeren uit Vogelenzang met het programma Diamantenjacht, dat reeds in ons vorig nummer geplaatst is. Hij krijgt hiervoor een printer.
De software-pakketten zijn gewonnen door: Gestel van Dimmen uit Kaatsheuvel, C.van de Meyden uit Den Helder, Jeroen Jagt uit Hengelo en Pieter Delu Knol uit Kampen.

In de komende nummers zullen we deze listings publiceren.
De prijswinnaars hebben reeds bericht gehad.

Redactioneel

Dit is alweer het laatste nummer van 1986.
In het nu bijna afgelopen jaar zijn er nogal wat woelingen geweest op de internationale computermarkt. De computers voor zakelijk gebruik rukken steeds verder op en af en toe leek het alsof de home computers terecht kwamen in de hoek waar de klappen vallen.

Wat Commodore betreft is de balans in een min of meer wankel evenwicht. De PC-10 en daarvan afgeleide „hogere" versies handhaven zich heel behoorlijk in de groeiende MS-DOS markt.

De C-64 blijft één van de beter-verkochte low-end machines. De C-128 daarentegen lijkt niet echt door te breken, waarschijnlijk ook omdat deze computer een beetje tussen de wal en het schip is gevallen.

De Amiga lijkt zich een eigen plaats verworven te hebben als zakelijke computer met nadruk op grafische toepassingen. Commodore blikt dan ook met een gematigd optimisme de toekomst in. Wij zullen deze toekomst ook in het komende jaar nauwgezet blijven volgen.

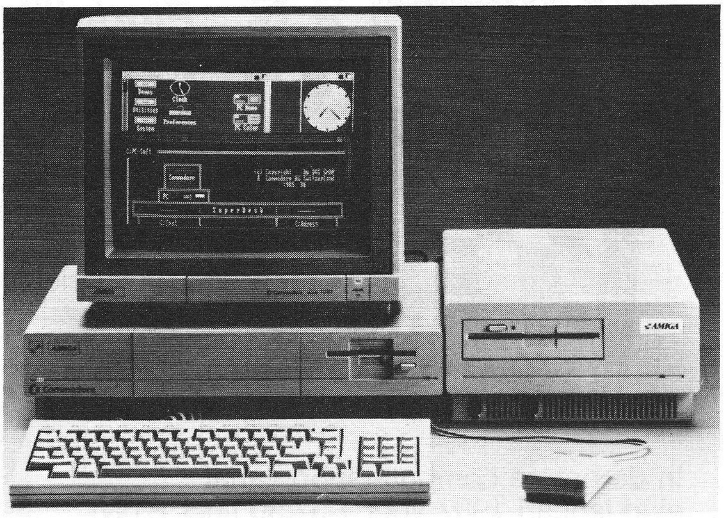
Voor het echter zover is wensen wij u prettige feestdagen en een voorspoedig begin van 1987 toe.

Luc Sala

Met de feestdagen voor de deur, heerst er bij Commodore in de Verenigde Staten een redelijk opgewekte stemming. Luc Sala bericht uit de VS.

Optimistische vooruitzichten

Commodore voorzichtig 1987 in



Op de Comdex in Las Vegas zochten we tevergeefs naar Commodore. Nu was dat bedrijf niet het enige, dat het af liet weten, ook Apple liet zich daar niet zien. Het was daarom vrijwel een totale PC-beweging, met veel 80386 nieuws en laserprinters.
Toch viel in de wandelgangen nog wel wat nieuws te vernemen over Commodore. het bedrijf zelf is niet ontevreden met de verkopen van de computers en de Amiga blijft voorzichtig aan doorlopen. Men verwacht binnen een paar maanden wel weer een upgrade voor de Amiga uit te brengen, waarover hieronder meer.

De laatste maanden, na een vrij slechte zomer, gaan de softwareverkopen weer wat lopen. De grote softwarehuizen zijn weer wat optimistischer en hebben hele reeksen nieuwe produkten klaarliggen. We spraken met Brad Fregger, de producent van vele Activision toppers zoals Hacker en Music Studio en hij tipte het adventure **Aliens** en een **Transformer** spel met Japanse inslag van David Crane. Brad is juist voor zichzelf begonnen, dus was hij wat vrijer om over de marktsituatie te praten dan in zijn Activision tijd. Volgens hem is er weer veel behoefte aan nieuwe spelletjes, de winterslaap van de software- uitgevers, die een zwakke tijd hebben gekend, is weer over. De meeste grote bedrijven, dat zijn Electronic Arts, Epyx en Activision zijn heel tevreden, al heeft Activision moeite om eindelijk weer eens winst te maken. Het bedrijf heeft in de videogame jaren gouden winsten gemaakt, maar sprong daar ook ruim mee om.

America's Cup

Electronic Arts is ingesprongen op de hysterie in de VS rond de komende America's Cup en een spel uitgebracht, dat onmiddelijk tot het officiële spel is uitgeroepen. In het algemeen brengt men nog steeds spelletjes uit voor de C-64 en C-128 en zeker op de komende Consumer Electronics Show in januari in Las Vegas kunnen we nog heel wat leuke dingen verwachten.



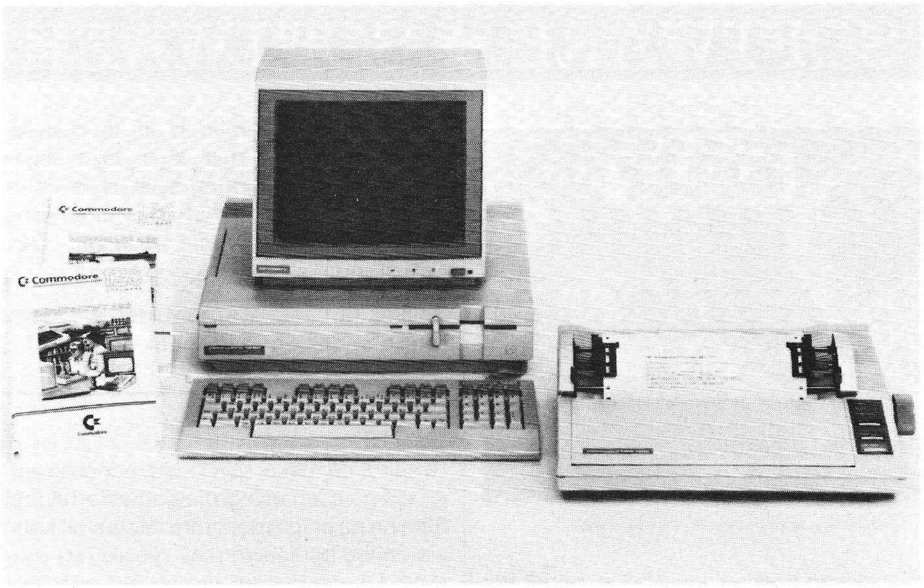
Thomas Rattigan, president van de Noordamerikaanse divisie van Commodore.

Resultaten Commodore

De balans van Commodore ziet er beter, maar nog niet echt florissant uit. En dat is ook moeilijk, de gecombineerde verliezen in de laatste twee jaar belopen maar liefst 241,8 miljoen dollar. Dat zijn fenomenale bedragen, zeker als we het aantal verkochte computers in aanmerking nemen. Schatten we dat op een kleine 2,5 miljoen stuks, dan kostte iedere computer, die de deur uitging CBM een paar honderd gulden. Ze hebben daar in Pennsylvania rare dingen uitgehaald, dat blijkt wel. Maar goed, het loopt nu allemaal weer en men heeft over het voorlaatste kwartaal zelfs weer een bescheiden winstje van 1,2 miljoen dollar gemaakt. En het afgelopen kwartaal ging het zelfs nog iets beter, met 3,7 miljoen winst. Maar ja, de banken houden CBM nog steeds in een ijzeren greep en eisen niet minder dan 32 miljoen dollar in 1987 en 90 miljoen dollar in 1988 terugbetaling op lopende leningen.
Clive Smith, die al vrij lang de marketing strateeg van Commodore was, is nu ook officieel verantwoordelijk voor de verkoop van de Commodore micro's.

lijk met een behoorlijk hogere prijs, maar het ziet er naar uit, dat men de Amiga dan ook veel meer IBM-compatibel zal gaan maken. Dat wil zeggen niet alleen een software-compatibiliteit, er komen dan ook PC-comptibele uitbreidingssleuven in de machine. Bij de PC worden deze sleuven, "slots", gebruikt om allerlei uitbreidingskaarten in te steken. Extra geheugen, interfaces voor video, RS-232, controllers voor extra diskdrives, men weet zelfs hele harde schijven op zo'n kaart te proppen. Dergelijke add-on kaarten zijn voor een belangrijk deel het geheim achter het succes van de IBM PC. Deze nieuwe Amiga, die mogelijk 2500 als indicatie krijgt, heeft 1 MB aan geheugen en werkt met de 68000 processor, maar mogelijk grijpt men ook naar de veel snellere 68020, dit om de machine in lijn te brengen met de nieuwe 80386 PC's, die nu beginnen uit te komen. In wezen gaat Commodore met deze aanpak een heel eind in de richting van IBM, met als extra voor de gebruiker dan de grafische en muzikale kwaliteiten van de Amiga chip-set. Of CBM hiermee van zijn geloof afvalt is eigenlijk niet meer echt an de orde, met name in Europa is men met de PC en AT toch al een zeer duidelijke kloon-weg opgegaan. ●

Luc Sala



C-128 Tekstsysteem.

Amiga als redding

Met het kerstseizoen voor de deur en wat aantrekkelijke kortingen als lokkers, ziet men de verkopen deze winter weer stijgen. Het moet de Amiga worden, die CBM weer op het paard gaat helpen. Vreemd genoeg zijn de hardware sales voor de Amiga bevredigend, maar loopt de software nauwelijks. Bij de Atari ST is dat net andersom, daar is de software de grote trekker. Volgens sommigen ligt dat aan de instelling van de Amiga koper, dat is een echte "hacker", die zijn eigen software schrijft. De ST koper is meestal een beginner, die wil software van de plank gebruiken.

Nieuwe Amiga's

Er zijn sterke geruchten, dat er een reeks nieuwe Amiga's gaat komen in de eerste maanden van het nieuwe jaar. Waarschijn-

Captain Kelly

In Captain Kelly, van het engelse Quicksilva, is de hoofdpersoon een huurling, die via de "Space Gazette" op een advertentie gereageerd heeft. Zijn doel is om de controle te krijgen over het centrale besturingscentrum van een geheimzinnig ruimteschip, dat door robots bestuurd wordt. Het ruimteschip is tamelijk groot en onderweg zijn zuurstoftanks, munitieopslagplaatsen etc. beschikbaar. Bovendien moeten alle robots uitgeschakeld worden, en om dat te bereiken komen er erg veel zweetdruppels en heel wat joystickwerk aan te pas.



Het beeld ziet er ook redelijk goed uit: de graphics zijn weliswaar niet 3D, maar toch goed ontworpen en kleurig. Hoewel de muziek van mindere kwaliteit is, houdt het spel de spanning er in, en daarom kunnen we u dit programma best aanraden. ●

Collapse

Firebird, de bekende "Elite" dochteronderneming van British Telecom, brengt het programma Collapse op de markt. Een spel uit de "Silver Range", dus goedkope



software van naar ons gevoel heel matige kwaliteit. Het gaat er om in een veld een soort koalabeertje alle spikkels op te laten eten, een beetje in de aloude Pacman traditie. Wanneer dit gebeurd is, gaat men door naar de volgende ronde. Dit programma is weer een van de vele in zijn soort: het geeft een tweedimensionaal beeld, waarin het een en ander opgegeten dan wel verzameld moet worden. De opzet en de eenvoudige uitvoering doen denken aan een spel uit de beginjaren van de C-64. Het enige dat de geringe kwaliteit nog goed zou kunnen maken is de lage prijs, 2 pond in Engeland. ●

Deactivators

Dit spel, op de markt gebracht door Ariolasoft, heeft weer de kwaliteit die we van dit softwarehuis gewend zijn. (andere bekende door Ariolasoft gedistribueerde games: Hacker I en II, Law of the West etc...). Dat de speler gebruik maakt van droids, dit zijn afstandbestuurbare robots, om zijn doel te verwezenlijken, is geheel volgens de laatste software-mode. Dit is ook het geval met recentere adventures zoals Hacker en De Kapriolen. Door deze robots handig te besturen moet men alle geplaatste bommen uit het gebouw zien te verwijderen. Zo gauw dit gebeurd is krijgt de speler een nieuwe taak: een groter gebouw met meer bommen en meer andere gevaren...

De spelsituatie wordt op het beeldscherm 3-dimensionaal door twee monitoren weergegeven. (al weer zo'n modeverschijnsel !). Onder de monitoren is een keuzemenu waar men kan kiezen welke robot gebruikt wordt en wat hij moet doen (de speler beschikt over meedere robots tegelijk). Aangezien de enige mogelijkheid om de bommen onschadelijk te maken is, ze van de ene naar de andere robot over te gooien (terwijl ze op scherp staan), is het te begrijpen, dat de speler heel wat spannende uren en zweetdruppels in dit programma zal moeten investeren om tot resultaten te komen. Al met al is deze actieadventure een programma dat bij de rechtgeaarde spelfanaat niet in de verzameling mag ontbreken. Van Ariolasoft voor ƒ 69,90 op disk en ƒ 45,- op tape. ●

Beslist het leukste spel dat wij deze ronde onder ogen kregen.

Druid

Firebird, de maker van Druid, scoorde enige tijd geleden een grote software-hit met **Elite** en na enige recentere teleurstellingen waren we benieuwd naar dit nieuwe produkt. De naam zegt het al: het gaat hier allemaal om een druide (medicijnman uit het stenen tijdperk). Zijn taak is het om Belorn, het land waar alles zich afspeelt te redden van de indringers, door de interplanetaire poort te sluiten. Op zijn weg daarheen moet hij vele gevaren trotseren, in de eerste plaats spoken, vervolgens bovenmaatse torren en nog veel meer engerds. Zijn taak is dan ook haast onuitvoerbaar en vele voorgangers, die het eerder probeerden, waren nu als geesten rond. Gelukkig wordt hij af en toe bijgestaan door een hulpje en is het mogenlijk op bepaalde plaatsen extra (levens-)kracht in te slaan. Jammer genoeg wordt alles tweedimensionaal weergegeven en dat maakt Druid tot slechts een 'gewoon' spel. De kwaliteit is echter toch weer beter dan sommige andere programma's die we de laatste tijd van Firebird kregen.

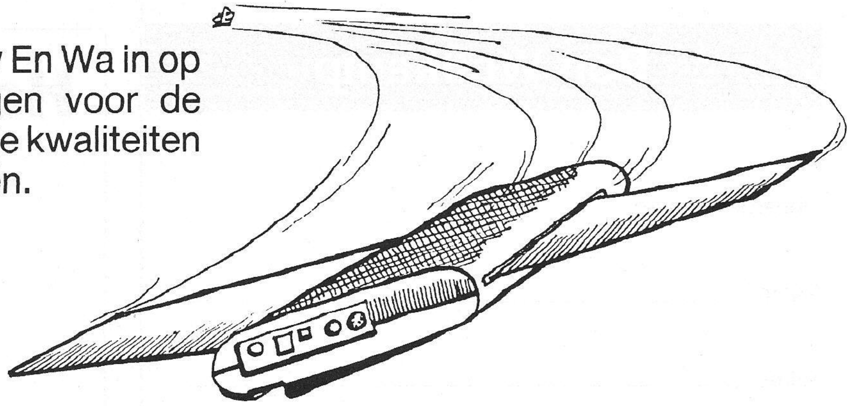
De prijs is ook zeer redelijk: ƒ 39,90 op cassette.

Nederlandse distributie Ariolasoft ●

Actiesoftware

Er komen een aantal hele goede spelletjes aan, waarbij de men de grafische kwaliteiten van de C-64 opnieuw tot een hoger niveau heeft opgepoetst. Daarbij is de disk nu vrijwel altijd onmisbaar, men haalt steeds weer nieuwe "graphics" en andere informatie van de disk af gedurende het programma. Van Accolade (importeur Aackosoft), bekend door o.a. HardBall, is er nu een vluchtnabootser, die **Ace of Aces** heet. Gebaseerd op het vliegen van een WOII jager/bommenwerper van de RAF, zit er een heel verhaal vast aan de simulator. Men kan verschillende opdrachten uitvoer, compleet met "briefing", keus qua wapens en verschillende gevechtssituaties. Bommenwerpen, luchtgevecht, raketten onderscheppen, alles in een fraaie grafische opzet. Een nieuwe vliegtopper. ●

In een aantal artikelen gaat P. Sjauw En Wa in op verschillende hardware-uitbreidingen voor de 64. Dit keer de diskversnellers, om de kwaliteiten van de disk drives wat op te voeren.



Diskversnellers

Volgens het Duitse blad CHIP van blijkt de CBM-64 nog steeds de meest verkochte homecomputer te zijn. Meestal in combinatie met de VC-1541, de betaalbare disk- drive van Commodore.

Helaas blijft er van het oorspronkelijke enthousiasme over deze drive weinig over nadat men hem enige tijd in gebruik heeft gehad. Met name de data-overdrachtssnelheid van de VC-1541 bezorgt een normaal mens een "bepaalde" aandoening die niet door een psychiater te genezen is. De enige remedie hiertegen is de aanschaf van een DISKVERSNELLER.

Gevaarlijk

Een waarschuwing vooraf is hierbij op zijn plaats, daar dit "geneesmiddel" ook door kwakzalvers wordt aangeboden. Hieronder volgt een lijst van bedrijven die volgens mij een verantwoord product leveren.

- Jann Datentechnik uit Berlin. Levert Prologic DOS (DM 300,-). In Nederland geleverd door Claessens Consultants in Breda (*f* 369,-).
- Christoph Dichte uit Brunsbuttel. Levert SpeedDos Plus (DM 269,-).
- RoBmoller uit Meckenheim. Levert Turbo Access (DM 295,-). Tevens Turbo Trans (DM 495,-).

Deze vier producten van Duitse firma's bestaan uit de volgende onderdelen:

- Nieuwe Dos.
- Nieuw Operating System voor de CBM-64.
- Parallelle data-overdracht.
- Ingebouwde handige routines.

Om de snelheid van de drive op te voeren beschikken o.a. SpeedDos en Turbo Access over een eigen disk formaat. Hierdoor worden de tijden voor de kopbewegingen aanzienlijk verminderd. Niet versneld worden programma's met een eigen laadroutine zoals bijvoorbeeld Flightsimulator II.

SpeedDos Plus

Dit wordt aan de User port aangesloten (niet doorgevoerd).

- Centronics interface op print.
- Formateren incl. verify : 23 sec.
- Laden van onder de oude Dos geschreven prg. gaat 6x zo snel. Onder SpeedDos 10x zo snel.
- Saven onder oude Dos gaat nu 1.5x zo snel. Onder SpeedDos gaat dit 10x zo snel.
- Interne disk-operaties worden 2-3 keer zo snel uitgevoerd.
- CP/M loopt 3x zo snel.
- Cassette routines zijn uitgeschakeld.
- RS 232-bus kan niet gebruikt worden (dus geen modem).
- Ingebouwde mini-monitor.
- Diverse handige routines.
- Er zijn verschillende copiers op de markt.

Er is zelfs een programma om SpeedDos 3x zo snel te maken.

Wat de compatibiliteit betreft valt het volgende op te merken. De oudere versie van SpeedDos bleek niet te werken met enkele nieuwere versies van DataBecker programma's. Deze nieuwe versie schijnt echter wat compatibiliteit betreft nogal verbeterd te zijn.

Turbo Access

- Wordt aan de Expansion port aangesloten (doorgevoerd).
- Centronics interface op print.
- Formateren in 16 sec.
- Laden onder oude Dos gaat nu 6x zo snel.
- Laden van programma in Turbo Dos formaat gaat 10x zo snel.
- Saven incl. verify onder Turbo Dos gaat 3x zo snel.
- Geen cassette routines meer.
- Diverse ingebouwde nuttige routines.
- Oude Operating system is m.b.v. een schakelaar weer in te schakelen.
- Copieerprogramma's worden bijgeleverd.
- Kostenloze update service.
- Compatible met bijna alle programma's.

Turbo Access is wat compatibiliteit betreft zeker beter dan SpeedDos.

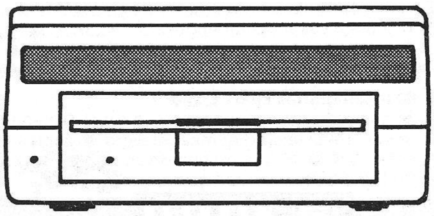
Prologic DOS

- Wordt aan de Expansion port aangesloten (doorgevoerd).
- Centronics interface op print.
- Formateren in 21 sec.
- Laden van programma's onder de oude Dos gaat 25x zo snel.
- Saven onder oude Dos gaat nu 12x zo snel.
- Andere disk-operaties gaan 2-3x zo snel.
- Men kan 40 tracks formateren. Hierdoor ca. 20 Kb extra .
- Er is een nieuw commando voor de VC-1541 bij gekomen.
- Men kan zowel hardwarematig als softwarematig naar het oude operating systeem terugschakelen.
- Geen cassette routines.
- Diverse ingebouwde nuttige routines.
- Zeer hoge compatibiliteit met de bestaande software.

Prologic Dos moet wat compatibiliteit betreft beter zijn dan de Turbo Access. Men kan nl. in 5 stappen softwarematig op het oude operating system overschakelen. De grote snelheid van dit systeem is onder meer te danken aan het feit dat steeds een hele track gelezen wordt in het RAM geheugen van de floppy. Nog sneller wordt het als we een hele schijf ineens kunnen lezen.

Extra geheugen

Turbo Trans is zo'n systeem, dat het probleem van de trage floppy op een in wezen bekende manier heeft opgelost. Er wordt beter gebruik gemaakt van het feit dat de VC-1541 een eigen processor heeft. Om een hele schijf in een keer in te kunnen lezen wordt het floppy geheugen met 256 Kb uitgebreid. In 10-20 sec. wordt dan de hele schijf ingelezen. Alle verdere disk operaties vinden dan in het RAM geheugen van de floppy plaats. Wijzigingen worden d.m.v. een interrupt op de schijf geschreven. Dit alles levert een overdrachtssnelheid van ca. 50 Kb/sec. op. Dit is ca. 120x sneller dan normaal. Het systeem wordt incl. diskmonitor en kopieer programma's geleverd.



Voor het maken van een leuk programma heb je soms tekens nodig die niet standaard in de tekenset van de computer zitten. Je moet dan dus zelf tekens ontwikkelen. Maar hoe werkt dat? Henk Boxma geeft in dit artikel de antwoorden.

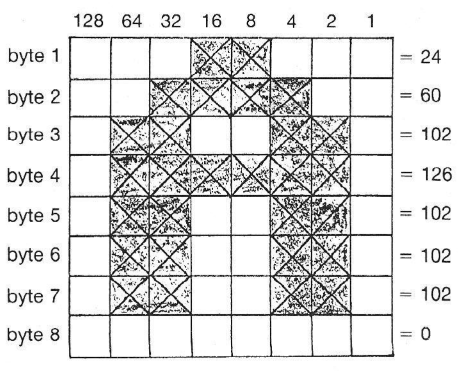
Voor C-16/Plus 4

Zelf tekens ontwikkelen

De karakterset (dat zijn alle letters, cijfers en symbolen) waarover de C-16 beschikt, bestaat uit 512 verschillende tekens. Die tekens zijn verdeeld over 4 hoofdgroepen. Je kunt echter maar over twee van die vier tegelijkertijd beschikken, omdat de tekens diapositief en dianegatief zijn. Met andere woorden: je kunt maar over 256 tekens tegelijkertijd beschikken.

Bij de C-16 (en trouwens iedere 8-bitter) is een karakter opgebouwd uit een matrix van 8 bij 8 hokjes. Een horizontale reeks van 8 hokjes noemen we een BYTE. Een karakter is dus opgebouwd uit 8 bytes. Iedere groep van 8 bytes op zich is weer genummerd:
A heeft waarde 1
B heeft waarde 2
C heeft waarde 3
enz.

De grootst mogelijke waarde is 255. Deze waarden kun je vinden in het C-16 handboek. De tabel staat bijna achterin. Let goed op dat je wel de zgn. "poke tabel" neemt en niet die met de ASCII-codes! Als voorbeeld nemen we de letter A



In byte 1 hebben bit 8 en 16 de waarde 1. We zeggen nu: bit 8 en bit 16 zijn "aan". De waarde van de poke wordt nu 8 + 16 = 24. De waarde van de poke van byte 2 is dus 4 + 8 + 16 + 32 = 60 enz.

Karakter op het scherm

Om een karakter op het scherm te krijgen moeten we de volgende stappen doorlopen.

❶ Het **PRINT**-commando.
Voorbeeld: PRINT "A"
Er zal nu een A op het scherm worden gezet.

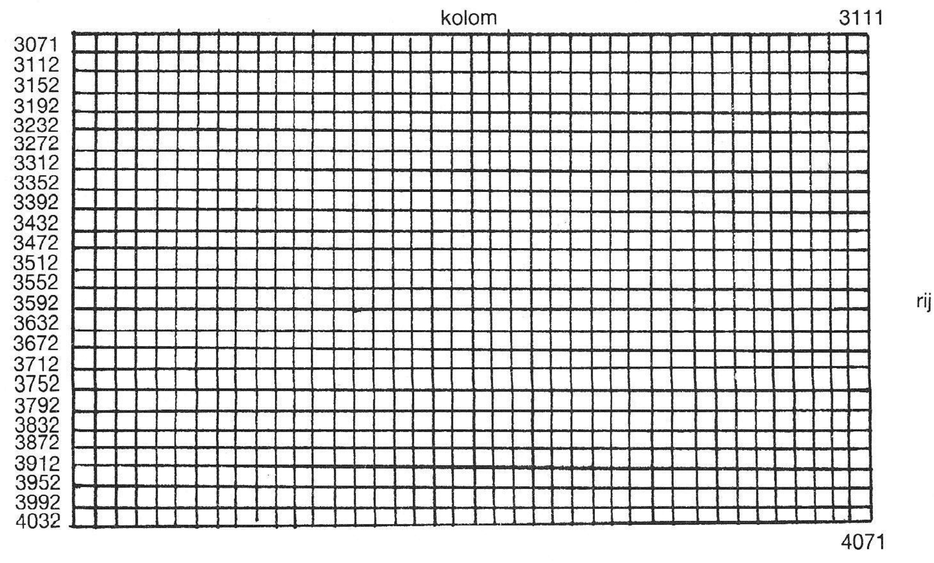
❷ Het **PRINTCHR$(x)**-commando.
Ieder teken op de C-16 heeft een ASCII-waarde, die ligt tussen de 0 en de 255. Let wel: deze waarden zijn niet gelijk aan de "poke"-waarden! Bijna achterin het handboek staan al deze waarden vermeld.
Voorbeeld: PRINTCHR$(65)
Er komt nu een A op het scherm, omdat die waarde 65 heeft. Vul in plaats van 65 maar eens een ander getal in tussen de 0 en de 255. Je zult zien dat iedere waarde zijn eigen teken heeft.

❸ Het **POKE**-commando.
Het beeldscherm van de C-16 is opgebouwd uit 320 bij 200 puntjes. Dat is gelijk aan 40 bij 25 hokken. Ieder hokje heeft zijn eigen adres. Het hokje linksboven heeft waarde 3072 en het hokje rechtsonder heeft waarde 4071. De overige hokjes hebben een waarde tussen 3072 en 4071 op de volgende manier:

Voorbeeld:
POKE4032,1 zet een A op adres 4032 (linksonder)
POKE3072,2 zet een B op adres 3072 (linksboven)
POKE3111,4 zet een D op adres 3111 (rechtsboven)
enz.

Kort samengevat:
Bij een POKE-commando worden dus achtereenvolgens het adres (0- 65535) en de code (0-255) aangegeven, gescheiden door een komma: POKE adres,code
Deze waarden staan in de poke-tabel.
☛ Nog een andere manier is het **CHAR**-commando.

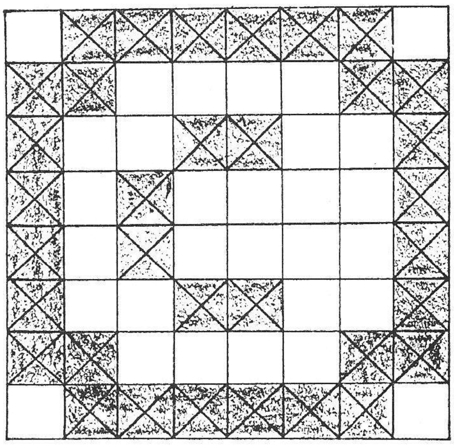
Voorbeeld:
CHAR kleurcode,x,y,string,reverse
kleurcode = 1 is multicolor, 0 is normaal kleur
x = teken rij 0 - 39 (van links naar rechts)
y = teken rij 0 - 24 (van boven naar beneden)
string = een naam tussen aanhalingstekens
reverse: 0 is uit (reverse off),
1 is aan (reverse on).



CHAR1,10,4,"test": zet het woord test in kolom 4 en rij 10.
Dit commando kan zowel in de grafische als in de niet-grafische mode worden gebruikt.

Een kort programma

Om een en ander van het bovenstaande in de praktijk te gebruiken, gaan nu zelf een teken ontwerpen. In onderstaand voorbeeld is dat een copyright-teken.
We tekenen eerst het schema en vullen dat in. Daarna kunnen we de waarden van de pokes bepalen en die erachter schrijven.



De volgende stap is een programma te schrijven om de karakters vanuit de ROM (het vaste geheugen) in de RAM (het geheugen dat we kunnen programmeren) te zetten. Dat doen we met dit programma:

```
10 v=65280
20 pokev+18,peek(v+18) and 251
30 pokev+19,peek(v+19) and 3 or 48
40 fort=832 to 849
50 read a : poke t,a; next:sys832
60 data 162,0,189,0,208,157,0,48,
189,0,209,157,0,49,202,208,
241,96
```

Standaard luiden de regels 20 en 30 resp.: poke 65298,196, en poke 65299,208
Tip: programmeer van tevoren een zgn. key op de volgende manier:
key1,"poke 65298,196:poke 65299,208" +CHR$(13)
Als je door een foute handeling (bv. een tikfout) toch een warbeeld krijgt, dan hoef je alleen F1 in te typen. Het scherm krijgt dan weer een normaal aanzien. Typ dan "HELP" en de computer zal de foutmelding op het scherm zetten.

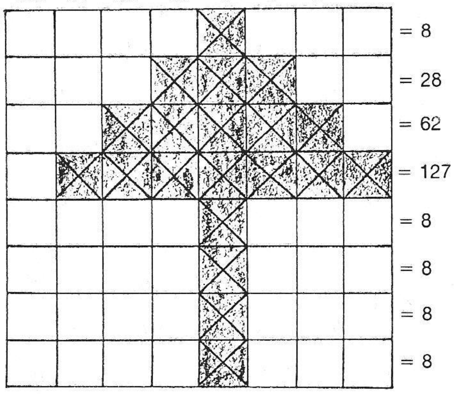
De regels 40 t/m 60 vormen een machinetaalroutine die de karakters vanuit de ROM in de RAM zet. Deze karakters starten op adres 12288.
Als ik karakters ontwikkel, begin ik op adres 12800, zodat ik alleen de symbolenset verander en niet de belangrijke letters en cijfers.

We vervolgen ons programma:

```
200 fort=12800 to 12800+(ak X 8)-1
210 read a: ifa> -1 then poke t,a : next
```

In regel 200 staat de variabele "ak" voor "aantal karakters". In ons geval is dat 1, omdat we 1 karakter hebben ontworpen en dat in het geheugen willen zetten. Regel 200 wordt dus:
200 fort=12800 to 12800+(1 X 8)-1
Nu zetten we de data van ons copyright-teken in regel 1000. Deze regel ziet er zo uit:
1000 data 126,195,153,161,161,153,195,126
Nu kunnen we het programma RUNnen. Er gebeurt niets zichtbaars, maar als we nu POKE3072,64 intikken zien we in de linker bovenhoek ons teken staan!

We maken nu nog een teken, een pijl. We tekenen weer een schema en vullen het in.



Het aantal karakters is nu dus 2. Daarom passen we regel 200 weer aan:
200 fort = 12800 to 12800 + (2 X 8) - 1
Nu moeten er nog wat data bij, die zetten we in regel 1010.
1010 data 8,28,62,127,8,8,8,8,-1
Ons programma is af. Als we het nu RUNnen en daarna POKE3072,65 intikken zien we ons pijltje in de linkerbovenhoek staan.
Sneller gaat: PRINT "-" : REM. Dit is shift + * teken.

We kunnen nu dus zelf tekens ontwikkelen, ze in het geheugen en vervolgens op het scherm plaatsen.
Je kunt deze informatie voor nog veel meer doeleinden gebruiken. Wat dacht je van een eigen spijkerschrift? Je hoeft dan alleen het adres in regel 200 aan te passen. Adres 12800 wordt nu 12288!

Veel success!

Henk Boxma



LUC SALA'S DATAKOLOM

Een bezoek aan de Comdex betekende dit jaar meer dan een handvol nieuws. Er was voor het eerst sinds jaren weer een positieve ervaring bij de pers en bij de bezoekers, er leeft weer iets. Maar wie bepalen nu eigenlijk wat in is, wat uit en wat er werkelijk aan de hand is.

Smaakmakers '87

Voorspellen welke richting de computerindustrie uitgaat is natuurlijk een soort gokspel. Niemand weet het zeker, alleen hebben journalisten daarbij het voordeel, dat ze mede die richting bepalen door een soort "self fulfilling prophesy". Dat klinkt tamelijk aanmatigend, maar laat ik uitleggen hoe dat werkt en waarom er meer dan een grond van waarheid in zit.

Op de Comdex in Las Vegas kwamen dit jaar 82.000 bezoekers gemiddeld 3 dagen een volstrekt onoverzichtelijk aanbod aan microcomputers, randapparatuur, software en "vapourware" in ogenschouw nemen. Door de amerikaanse manier van rekenen, gebaseerd op het aantal uitgegeven badges, lijkt dat in vergelijking met bijvoorbeeld de Efficiency Beurs niet veel. Maar dat zijn dan wel bezoekers, die er minstens een vliegreis en een paar overnachtingen voor over hebben om daar in Vegas bedolven te worden onder de stroom nieuwtjes en pseudo-nieuwtjes. Ze komen er ook om hun collega's, leveranciers en klanten te ontmoeten en misschien ook wel om wat geld te vergokken.

Samen vormen ze de bovenlaag van de computerindustrie. Ze zijn meestal heel goed op de hoogte van wat er zich op hun terrein afspeelt. Dat komt omdat ze natuurlijk hun eigen produkten kennen, naar zo'n beurs komen om te kijken wat de concurrent er van bakt en ook veel horen van hun collega's.

De pers speelt hierbij een heel vreemde rol. De bijna duizend journalisten en mediamensen hebben er, net als de verkopers, in wezen een soort jaarlijks congres. Voor degenen, die wereldwijd kijken, is het meer een halfjaarlijks gebeuren, de Hannover Cebit Messe is ook een heel belangrijke beurs. In ieder geval is zo'n show de gelegenheid om eens te horen wat men er allemaal off-the-record van vindt en flink te roddelen. En het leuke is, dat de computerindustrie dat middels een eindeloze reeks persconferenties, persrecepties, hospitality-suites en feesten ook nog allemaal mogelijk maakt. Er gaat geen avond voorbij of je moet uit een heel schema kiezen en goed uitzoeken in welke hotels de beste parties zijn. Gezellig, want langzamerhand ken je de collega journalisten en ook de topmensen van de bedrijven redelijk goed. Het lijkt op een hele week feesten en afgezien van de vermoeide voeten en gebroken ruggen door het gesleep met dozen persberichten, lijkt dit een ideale job. En zo wordt het door de columnisten als John Dvorak van PC Magazine en Jerry Pournelle van Infoworld ook gebracht, ze rapporteren uitgebreid over dergelijke feestjes. Of ze het ook dit keer in die bladen zullen hebben over de Insider en de Insider Insider parties, weet wel dat ook dit jaar de vaderlandse journalistiek er weer bij was. Met een paar gewaardeerde collega's hebben we die befaamde columnnisten uiteindelijk zeer vroeg in de morgen in hun taxi's gezet. Op dat feestje was iedereen, die meetelt, dus mensen als Philippe Kahn, Leonard Tramiel, Lee Felsenstein, Stewart Alsop, Jan Lewis, enzovoorts.

Wat is nu toch het belang van al dat praten en roddelen van die veertig of vijftig "spielmachers" op zo'n beurs? Beïnvloeden ze nu werkelijk wat er gebeurt of zijn we met zijn allen alleen lachwekkend bezig. Ik weet wel zeker, dat al die "executives", de top-journalisten en -analisten en andere goeroes daar uiteindelijk gewoon hard aan het werk zijn. Hoe werkt dat. Na drie dagen rondrennen over de beurs, persconferenties, salesmeetings en forumdiscussies, heeft iedereen wel zo'n beetje de maximaal opneembare hoeveelheid gegevens geïnhaleerd. Een mens wordt na 30 keer bekijken van een 80386 de eenendertigste keer echt niet veel wijzer meer.
Gegevens, let wel, nog geen informatie. Het doorwerken van een enorme stapel persberichten over 386 micro's, het bekijken van 30 EGA-kaarten, demo's doorstaan van tientallen nieuwe softwarepakketten is bepaald geen leuk werk. Maar na een paar dagen is dat toch gebeurd en begint zich een mening te vormen over wat er nu eigenlijk op zo'n beurs gebeurt, over de trends en onderstromen. Maar ja, wat de eenling denkt kan natuurlijk wel eens falikant mis zijn. Je kunt wat gemist hebben, net die ene persconferentie vergeten zijn of het gewoon niet begrepen hebben. Maar voordat je als journalist, maar ook als baas van een groot bedrijf nu teruggaat naar je gewone omgeving, moet je natuurlijk wel proberen uit te vinden, of je eigen impressies wel kloppen. Dat kun je niet al te duidelijk doen, want er is niets vervelender dan er openlijk falikant naast zitten. Maar er moeten wel weer artikelen over geschreven worden of produkten bedacht worden. Stel je voor, dat niet EGA, maar PGC als de nieuwe grafische standaard van de grond komt.

Iedereen loopt dus na een paar dagen met een half-klare opinie, met een heleboel feiten, maar nog geen complete visie en zoekt dus naar bevestiging. En dat gebeurt nu op dergelijke parties, waar de echte smaakmakers aan elkaar snuffelen, elkaar uitdagen, roddelen, vissen naar bevestiging en in het algemeen toch tot een soort consensus komen. Dat zullen ze niet erkennen, men is tenslotte vreselijk trots op de eigen onafhankelijke mening, maar het gebeurt wel. En iedereen weet dat ook en probeert die gemeenschappelijke opinie ook wel te beïnvloeden. Want wat hier aan roddels en verhalen de ronde doet, duikt onherroepelijk een keer op in de publiciteit of in de directiekamers van de computerbedrijven. De opinieleiders van de industrie maken namelijk, informeel en zonder dat nu allemaal bij de naam te noemen, uit wat zij als de trends en hoofdlijnen van deze Comdex en daarmee van de industrie voor de komende 6 tot 12 maanden zien. Ze gaan daarmee de volgende dag aan de gang, schrijven artikelen en rapporten, en die mening sijpelt dan de komende maanden langzaam door in redacationele kolommen van kranten, bladen en andere publikaties. Dat duurt soms heel lang, de zaterdag na de Comdex brengen de NY Times, de San Jose Mercury News, en de San Francisco Chronicle al hun verslagen, maar ook de februari-nummers van veel maandbladen brengen nog Comdex nieuws. En raadt eens, wie in vrijwel al die bladen als columnist schrijven, jawel, dezelfde smaakmakers, die op al die parties rondliepen.

Misschien zijn we als journalisten beperkt in ons inzicht en kan men de Comdex Insider parties afdoen als industriële navelstaarderij zonder enig belang voor de werkelijke wereld. Ik denk het niet, ik heb het gevoel, dat daar op enig niveau het soort verfijnde afstemming plaats vindt, waarvoor we zelfs met electronic mail en 80486 processors nog geen "expert"-systeem kunnen bedenken. ●

L.Sala

TEST:

Floppy Flash Professional

Gima Print Service brengt een inbouwpakket op de markt, waarmee het mogelijk wordt 45 maal sneller te laden en 25 maal sneller te saven met de 1541 diskdrive. De 1541 is berucht om zijn traagheid: laden of saven van een taperecorder met een snellader gaat sneller dan werken met een 1541 diskdrive. Daarom zal deze module voor velen een grote verbetering van hun computersysteem zijn.

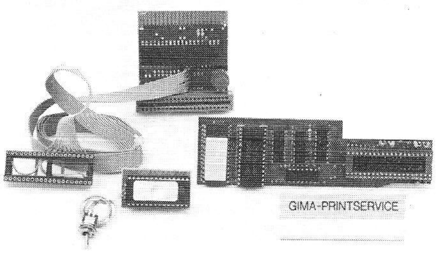
De oorzaak van de traagheid van de 1541 ligt in het feit dat het een seriële diskdrive is, waarbij de gegevens achter elkaar verzonden worden, in plaats van een parallelle, waarbij de gegevens tegelijkertijd verzonden worden. Een tweede oorzaak van de lage snelheid is de langzame Commodore DOS. De Floppy Flash Professional verhelpt beide problemen: de diskdrive krijgt een parallelle aansluiting op de cartridgepoort en de DOS wordt aanzienlijk verbeterd en versneld. Om dit mogelijk te maken worden in de computer en in de diskdrive IC's bijgeplaatst en vervangen.

De inbouw

Het inbouwen van het pakket vergt ongeveer een half uur. Hiervoor is geen speciale kennis vereist, het is gewoon een kwestie van het oude IC eruit en het nieuwe IC erinklikken. In de handleiding staat duidelijk hoe en wat er moet gebeuren.(Het vervangen van de IC's moet natuurlijk wel voorzichtig gebeuren.) Bij het aansluiten van alle kabels na het monteren komt er nu ook een parallel-kabel met uitgang op de cartridgepoort. Verbaas U er niet over dat ook de seriële kabel moet blijven zitten: de computer is dan dus met twee kabels aan de drive verbonden.

Naast het sneller laden biedt de Floppy Flash Prof. de gebruiker nog een aantal extra functies:

1. er kunnen 40 tracks geformatteerd worden i.p.v. 35. Er is dus meer ruimte voor software op een diskette;
2. de computer geeft aan waar in het geheugen geladen wordt;
3. op het printplaatje dat achter in de computer gestoken wordt zit een resetknop, waarmee verschillende soorten resets mogelijk zijn;
4. er zijn extra (of snellere) DOS commando's. Deze kunnen met de functietoetsen opgeroepen worden;
5. de gebruiker beschikt over een aantal extra cursormogelijkheden. Een ervan is het vermelden zeker waard: iedereen heeft wel eens in de knoop gezeten met ongewenste cursorbesturingstekens na het aanhalingsteken. Met CTRL/spatie is dit nu direct uit te schakelen.

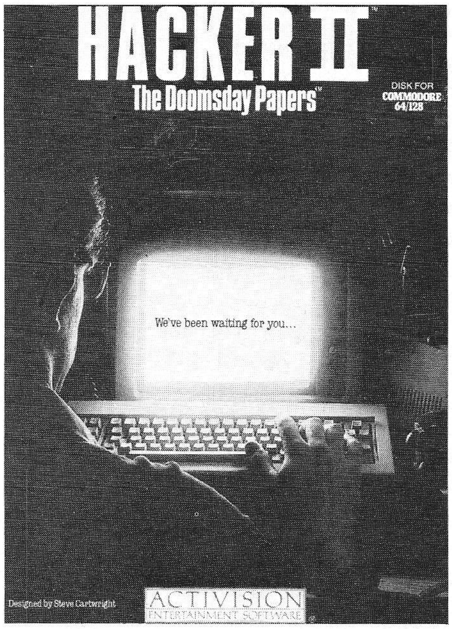


Aanrader

Wij zijn zeer onder de indruk van de prestaties van deze uitbreiding voor de C-64. De naam Floppy Flash **Professional** is zeker terecht, want hij geeft uw diskdrive professionele eigenschappen. Binnenkort komt er ook een versie voor de 1541C en de 1570/1571 in combinatie men een C-128.
Het pakket wordt geleverd in combinatie met een diskette met programma's. De prijs van het systeem is ƒ 325,- en voor mensen die al Speeddos of Floppy Flash bezitten is de prijs voor alleen de uitbreiding ƒ 199,-. Deze prijzen lijken wat hoog maar zijn naar onze mening zeker gerechtvaardigd.
Wanneer de Floppy Flash is ingeschakeld (dit gebeurt met een schakelaartje) kan er niet met tape gewerkt worden.

Hacker II

Het softwarehuis Activision, in ons land geimporteerd door Ariolasoft, houdt van succesnummers. Maar men wil ook graag een stukje creativiteit behouden. Hacker II (van Activision) is daarom geen vervolg op de beroemde krakeradventure Hacker I van hetzelfde softwarehouse, zoals men zou verwachten. Zo wordt in tegenstelling tot dat spel bij Hacker II wel een handleiding geleverd. (die is ook in het Nederlands).
Hierin staat in het kort de bedoeling en de werking van het programma:
Het doel van de speler is, om uit een geheim militair complex in Siberie een kopie van de z.g. "**Doomsday papers**" op te halen. Deze zeer geheime documenten zouden het de Sovjet-Unie mogelijk maken al de Westerse regeringen omver te werpen. Om deze papieren te bemachtigen en zo de vrije wereld te redden, hebben agenten van de C.I.A. drie MRU's in het complex geplaatst. (Mobile Remote Units zijn afstandbestuurbare robots.) Deze robots zijn vanaf de huiscomputer te



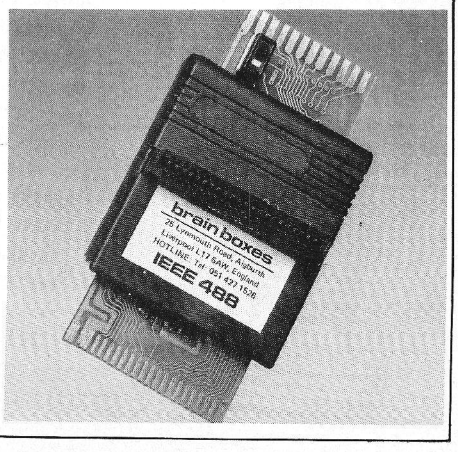
bedienen. Op het scherm wordt het spel weergegeven door 4 monitoren en een besturingspaneel. Met de monitoren kan men het camera-beveiligingssysteem aftappen of op een kaart van het gebouw zien waar de robot is. Het moeilijkste aan het spel is te zorgen onontdekt te blijven. Hulpmiddelen hiervoor zijn de wachtpatrouille- en cameradetector. Als de robot eenmaal ontdekt is, wordt hij vroeger of later (maar onherroepelijk) vernietigd door de Annihilator. Maar de makers van Hacker hebben weer lang niet alles verteld. Gedurende het spel zult u meermalen voor (on)aangename verrassingen komen te staan.....

Goede graphics

Op de uitvoering van Hacker II is weinig aan te merken: de graphics zien er leuk uit en zijn geanimeerd. De geluidseffecten zijn goed en het programma is in geen enkel opzicht saai te noemen. Het enige nadeel van deze beeldadventure is de zaak tamelijk ingewikkeld is opgebouwd. Hacker II kan niet "even op een middagje" goed worden gespeeld. Dat is voor de kenners dus een aanbeveling.
Wij kunnen daarom degenen die geen bezwaar hebben tegen wat denkwerk en ook tijd over hebben, dit programma van harte aanbevelen, het is een beeldadventure met pit. ●

IEEE 488 interface

Het Britse bedrijf Brainbox speelt in op de vraag naar een IEEE interface voor de C-64 en C-128. Tot de belangrijkste IEEE-randapparatuur behoren de snelle disksdrives van Commodore. Bezitters van de 4040, 1001, 8050 en 8250 drives of een Commodore hard disk kunnen zo de trage seriële databus omzeilen. Een tweede belangrijke toepassing van het 488 interface is de mogelijkheid om Commodore en andere IEEE-printers, deze afdrukkers zijn incompatibel met de C-64 en C-128 machines, op een Commodore huiscomputer aan te sluiten.
Het IEEE 488 interface van Brainbox wordt in het cartridge slot van de C-64 of C-128 geplugd. De vooruitdenkende ontwerpers hebben het interface aan de bovenzijde van een nieuwe socket voorzien zodat reeds bestaande uitbreidingmodules zoals snelladers en ROM-cartridges tegelijktijd met de IEEE 488 gebruikt kunnen worden.
Uit het interfacekastje komen aan de vooren achterzijde de bekende groene printplaatuiteinden met de verbindende metalen kontaktstrips. De voorste kontaktstrips worden in het cartridgeslot geplugd. Het achterste printplaatje, met C-64/C-128-schakelaar, is de eigenlijke IEEE-connector. Deze eindconnector wordt via een **Pet-to-IEEE**-kabel met de desbetreffende IEEE-randapparatuur verbonden.
Het Brainbox IEEE 488 interface biedt de Commodore C-64 en C-128 bezitter de volgende mogelijkheden:
☛ De hoogst mogelijke dataoverdrachtssnelheid bij IEEE-diskdrives. Na het commando FAST werkt de C-128 op zijn hoogste kloksnelheid. NB.: na het FAST-commando valt de RF-output tijdelijk uit.
☛ CP/M-compatibiliteit.
☛ Omschakelbaar tussen de C-64- en C-128-mode. Deze optie is zowel belangrijk voor het draaien van software als voor het gebruik van de verschillende Commodore computers.
☛ Een aanzienlijk grotere printerkeuze.
☛ In de 64-mode is het hulpprogramma DOS Support standaard beschikbaar.
☛ Met behulp van meerdere Brainbox interfaces kan een relatief goedkoop Commodore huiscomputernetwerk worden opgezet. De verbinding wordt via een 18 pins lintkabel tot stand gebracht.
☛ Seriële en parallelle randapparatuur kan gelijktijdig en door elkaar gebruikt worden.
Voor de beziters van de modernere 1571 en 1572 Commodore drives vervalt het voordeel van een sneller dataoverdracht. Beide drives maken immers gebruik van een veel snellere databus dan het oude seriële busontwerp.
Het Brainbox IEEE 488 kost inclusief het 27 pagina's tellende Engelse manual ca. *f* 280,-. Nadere info bij uw computershop of Brainbox, Lynmouth Road, Liverpool L17 6AW, ☎ Engeland, 051-2202500. ●



Bob Munniksma, zelf leerkracht, geeft wat aanwijzingen over het zelf met de computer aan de gang gaan in het onderwijs.

Zelf leerprogramma's maken in Basic

Je hoort wel eens zeggen: een leerkracht moet niet zelf programmeren voor onderwijsdoeleinden. Hij schrijft toch ook geen leerboeken. Er zit in deze bewering een zekere mate van waarheid. Op het eerste gezicht denk je, dat de bewering echt opgaat.

Toch schuilt er, vanuit de onderwijspraktijk gezien, een onwaarheid in. Leerkrachten schrijven wel degelijk leerboeken. Al worden die in de vorm van een boek nooit officieel uitgegeven. Waar mijn redenering heen wil is duidelijk. Er zijn veel goede en bruikbare leermethoden op de leerboekenmarkt. Deze boeken worden vaak vanuit een bepaalde onderwijsfilosofie geschreven. En toch moet omwille van de commercie een zekere algemeenheid in de methodiek aanwezig zijn. Het boekenpakket moet ook nog eens verkocht worden.

Eigen materiaal

Hoe goed de boeken ook trachten aan te sluiten bij de onderwijspraktijk, er zijn toch manco's, die in de meeste gevallen dan door de leerkrachten die er mee werken, worden aangevuld. De vormgeving van die aanvullingen is overigens vaak eenvoudig, maar daarom niet minder doeltreffend: een pakketje stencils met de nodige oefen/leerstof. En om nu terug te komen op de redenering en er op voort te borduren: het is dus onzin dat leerkrachten niet zouden moeten gaan programmeren. Als een leerkracht met problemen zit t.a.v. bepaalde leerstof en hij of zij denkt daarvoor een computer te kunnen inzetten, is het juist zeer zinnig om dat stukje leerstof zelf om te zetten in een leerprogramma. De leerkracht zit aan de bron. Het probleem is geconstateerd, een analyse vaak al gemaakt en daarmee ook een mogelijke oplossingsmethode. Je bent als leerkracht met computerkennis dan toch wel onnozel, als je niet zelf aan de slag gaat. Werkbladen maken doe je toch ook zelf, als dat nodig is. En ook al zien de eigen produkten er nooit zo mooi uit als hun commerciele tegenhangers, ze zijn vaak effectiever en toegespitster op dat ene probleempje. Ook zijn ze natuurlijk veel minder kostbaar! Dat geldt voor boeken en software.

Toepassingsgebieden

Er zijn in het onderwijs een aantal toepassingsgebieden voor de computer. Dat zal voldoende bekend zijn. We richten ons hier slechts op een van de doelgebieden, nl. trainen, instrueren en toetsen; kortom de ondersteuning van de dagelijkse onderwijspraktijk. En om het overzichtelijk te houden gaan we dieper in op het trainingsaspect. Aan de hand van een programmavoorbeeld zal de aanpak van een programmaontwerp op dat gebied besproken worden.

Als je een onderwijsondersteunend programma wilt gaan maken, moet je een aantal gulden regels in acht nemen. Er is een logische volgorde in de aanpak. Vaak wordt er "in het wilde weg" geprogrammeerd. Resultaat is een vaak onoverzichtelijk, niet goed lopend programma. De leerkracht is dan zo vaak nodig in de praktijk, dat van ondersteunen niet eens sprake is. Vaak wordt het tegendeel van de doelstelling bereikt.

De aanpak

Ten eerste moet je weten welk onderwerp of deelvakgebied in aanmerking komt om via de computer te trainen. In ons geval is dat het automatiseren van de tafels van vermenigvuldiging.
Ten tweede zal er een doelstelling geformuleerd moeten worden. Wat wil je bereiken met het programma. Wat moeten de leerlingen na een bepaalde oefenperiode kennen of kunnen.
Dan moet je een analyse maken van wat je eigenlijk wil, dat het programma doet. Moet het alleen trainen of juist toetsen. Moet er ook ondersteuning zijn voor de leerlingen, als er problemen ontstaan. Moeten de resultaten bijgehouden worden of wil je ze zelfs op papier aan het einde van de dag. Je ziet een hele waslijst. Schrijf alles maar eens op, daar heb je later veel plezier van.
Een zeer belangrijk aspekt is de gebruiksvriendelijkheid. Er moeten tenslotte kinderen mee werken. Tussenkomst van de leerkracht moet eigenlijk niet nodig zijn. Vervolgens moeten we nog denken aan een goede dokumentatie van het programma, zodat ook in een later stadium het programma nog overzichtelijk is voor de programmeur. Ook al is dat niet de oorspronkleijke auteur. Dan kan op een bijbehorend papier. Slimmer zijn REM statements in het programma zelf. Zeker als het programma zelf weinig geheugenruimte nodig heeft. Tenslotte is er nog het aspekt van de vormgeving. Het oog wil ook wat. De grafische mogelijkheden van de Commodore familie zijn voldoende groot om er iets leuks van te maken. In het voorbeeld hebben we de beeldscherm lay-out bewust eenvoudig gehouden. Het gaat immers om de programmering in BASIC. Overigens houdt niemand je tegen om dat aan te passen.
Tot nu toe is de computer nog niet eens aan geweest. Dat zal nog een tijdje niet nodig zijn, want je kunt beter eerst een stroomschema maken, om later het programmeren te vergemakkelijken. In zo'n schema breng je alle zaken, die van belang zijn, in kaart.

Modulair of lineair

Bepaal nu of het programma modulair of lineair wordt opgebouwd. Dat betekent werken met afgeronde programmadelen, die via GOSUB worden geaktiveerd of dat alle delen van het programma achter elkaar komen. Gezien de BASIC van de Commodore computers is de keuze voor lineair programmeren aan te raden. In het programma voorbeeld is dit ook gedaan. Er worden wel een aantal sprongen gedaan. Deze maken het programma zeer wendbaar.

De opbouw van het programma

Nu gaan we het programma opbouwen. Bepaal eerst welke regelnummers gereserveerd worden voor de verschillende programmaonderdelen. Bekijk het overzichtje. We beginnen met de INITIALISATIE.
Hier worden variabelen benoemd en gedimensioneerd. Tevens wordt een set kaders opgebouwd, omdat die dan niet steeds geprogrammeerd hoeven te worden. Ze worden in de variabelen K1$ tot en met K7$ gezet. Even tellen, K1$-K4$ zijn elk 40 karakters lang. Een hele schermbreedte dus. Op regel 10 wordt de variabele NE$ gemaakt. We gebruiken deze om de tekst eenvoudig op de gewenste plaatsen te krijgen, omdat een commando zoals LOCATE ontbreekt in BASIC Versie 2.0. NE$ staat voor NEER. Vergelijk de CHR$-tabel in de handleiding van je computer. En omdat we het woordje RETURN veel in de schermtekst gebruiken, komt die ook in een variabele (zie regel 90).
Dan komt de instelpagina voor de leerkracht. Hier kan hij of zij bepalen welke tafel door het programma wordt gegenereerd en hoe lang elke leerling aan de computer mag werken. De computer accepteert geen 0 als invoer en ook geen letters (regels 130 en 135). Op regel 140 staat een wat toverachtige formule. De invoer van de tijd ging met de alfanumerieke variabele TD$. Deze vormt de basis voor de tijdsinstelling voor de leerling later in het programma. Daar wordt namelijk een vergelijking gemaakt met de tijd op de ingebouwde klok. Het formaat van TI$ (deze is vastgesteld) moet overeenkomen met het te vergelijken formaat van TYD$. Vandaar die regel 140.
Ook staat op deze pagina een soort menu voor de werking van de Functie-toetsen vanaf regel 145. Onthouden dus. Op regel 175 staat GOSUB 1500. De sub-routine op die regels wacht op het indrukken van de RETURN-toets.
Op regel 300 gaat het leerlingengedeelte van start. Hier geeft de leerling zijn of haar naam. De computer bewaart die naam met de nog komende gegevens van die leerling. Dan volgt vanaf regel 400 een hulppagina voor de leerling. Hij kan nu zien wat hij moet doen. Ook later, als hij bezig is en niet meer weet hoe het ook weer moest, kan een druk op de f 7 toets uitkomst brengen: de pagina komt dan terug, zonder de rest van de oefening te verstoren. Ook hier weer GOSUB 1500.

Het oog wil ook wat

Dan wordt het scherm opgebouw vanaf regel 500. De klok wordt op "000000" gezet en de K$ kaderstrings zetten het beeld op. De kaders zijn echter nog leeg en



krijgen door de regelnummers vanaf 600 een invulling. Zichtbaar worden de naam, de te oefenen tafel, het aantal gedane sommen, het aantal goede antwoorden en de score. Deze laatste is een soort beoordeling tijdens de rit. Regel 640 en 645 wissen de kadertjes van de tafelsom en de mededeling of het gegeven antwoord goed of fout was. Later zie je wel wat hiermee wordt bedoeld.
Dan komen de sommen op de proppen. Vanaf regel 700 wordt een tafelsom gegenereerd en op het scherm gezet in zijn kadertje. De meeste regels in dit programmadeel zijn om voor een juiste lay-out te zorgen. Dan springt het programma naar regel 1000. Daar wordt het antwoord ingevoerd. Vanaf regel 800 wordt het gegeven antwoord gecontroleerd. Vervolgens wordt gekeken of de tijdslimiet al is bereikt, de score wordt berekend en er wordt een passende tekst op het scherm geplaatst. Dan springt het programma terug naar regel 600 en alles gaat van voren af aan. Is de tijd voorbij, dan gaat het programma naar regel 2000, waar de leerling een overzicht krijgt van zijn prestaties en tevens het verzoek de volgende leerling te roepen. Drukt de leerling op RETURN, zoals gevraagd, dan start het programma opnieuw vanaf regel 300. Alle gegevens omtrent tafel, aantal gedane en goede sommen en de gekregen tijd zijn nu opgeslagen in het geheugen in de verschillende array's, die eerder werden gedimensioneerd.
Door tijdens het lopen van de oefening op

SHIFT/f5 te drukken kom je in die administratie. Steeds 20 leerlingnamen komen met hun gegevens op het scherm. Een druk op de spatie geeft de volgende 20. SHIFT/f1 start het programma vanaf de instelpagina voor de leerkracht. SHIFT/f3 start vanaf het leerlingengedeelte en f7 geeft de hulppagina voor de leerling. Al deze functietoetsen hebben dezelfde werking tijdens de oefening. Je kunt de oefening dus op elk gewenst moment onderbreken en opnieuw ergens in het programma starten.
De routine voor het invoeren van de antwoorden (regel 1000) behoeft speciale zorg. De gegeven oplossing is slechts een voorbeeld van wat mogelijk is. Nog lang niet perfekt dus. Uit de praktijk zal blijken of deze constructie voldoet.

Laatste detail

Hoewel het om getalleninvoer gaat, wordt er toch gewerkt met een stringvariabele. Dit is, omdat een mogelijke foutmelding bij het invoeren van een letter dan achterwege blijft. Wel wordt met de ASC-functie gekeken of een bepaalde speciale toets wordt ingedrukt. Bijvoorbeeld een functietoets. De DELETE-toets wordt genegeerd, omdat die het schermplaatje verstoort. Met de SPATIE kan een foute invoer worden teruggenomen. De antwoordstring AN$ wordt dan met een element verminderd. Bij het indrukken van RETURN wordt de invoer verlaten om naar regel 800 te springen.
Tot zover het voorbeeld-programma. Meer beoogt dit stukje BASIC ook niet te zijn: een voorbeeld. Het gaat meer om de gedachte, zoals ik die in het begin uitte. Als je klaar bent met programmeren volgt natuurlijk een fase van uittesten. Eerst droog en dan in de schoolpraktijk. Je staat er soms versteld van, hoe leerlingen het programma toch laten vastlopen. Weer aan het veranderen dus, tot alles goed loopt. En nu........aan de slag.

B.M.

PROGRAMMA-OPBOUW

0 -	INITIALISATIE
100 -	START LEERKRACHT
300 -	START LEERLING
400 -	HULPPAGINA
500 -	SCHERMOPBOUW
600 -	SCHERMINVULLING
700 -	SOMMENGENERATOR
800 -	ANTWOORDCONTROLE
900 -	TIJDSCONTROLE
910 -	SOMMENTELLER
920 -	SCOREBEREKENING
1000 -	ANTWOORD INVOEREN
1500 -	TOETS OPHALEN
2000 -	EINDE OEFENING
5000 -	ADMINISTRATIE

```
  5 rem tafel oefen programma
 10 n=100:rem aantal leerlingen
 15 dim n$(n),as(n),g(n),sc(n),td(n),tt(n)
 20 poke 53280,9:print chr$(14)chr$(8)
 25 :
 30 rem initialisatie
 35 ne$=chr$(19):fort=1to21
 40 ne$=ne$+chr$(17):next
 45 fort=1to38:k$=k$+chr$(96)
 50 sp$=sp$+chr$(32):next
 55 k1$=k$+chr$(96)+chr$(96)
 60 k2$=chr$(176)+k$+chr$(174)
 65 k3$=chr$(125)+sp$+chr$(125)
 70 k4$=chr$(173)+k$+chr$(189)
 75 k5$=left$(k2$,19)+chr$(174)
 80 k6$=left$(k3$,19)+chr$(125)
 85 k7$=left$(k4$,19)+chr$(189)
 90 r$=chr$(18)+" RETURN "+chr$(146)
 95 :
100 rem start leerkracht
105 print chr$(147);
110 print"Met dit programma is het
    mogelijk alle"
115 print"tafels te oefenen."
120 print
125 print"Alle nodige instellingen
    kun je nu doen.":print
130 print"Welke tafel      ";:input tt$
    :tt=val(tt$):if tt=0 then 100
135 print:print"Hoeveel minuten ";
    :input td$:if val(td$)=0 then 100
140 tyd$=left$("0000",4-len(td$))+td$+"00"
145 print:print"Betekenis van de F-toetsen:"
150 print:print"f 1  = Naar de instelpagina"
155 print:print"f 3  = Naar de start
    van de oefening"
160 print:print"f 5  = Naar de administratie"
165 print:print"f 7  = Naar de hulppagina"
170 print:print"f-toetsen indrukken met SHIFT"
175 gosub 1500
180 :
300 rem start leerlingen
305 g=0:as=0:sc=0:x=x+1:print chr$(147);
310 print"Hoe is je naam : ";:inputn$
315 print chr$(147)"Hallo, "n$
320 print
325 print"Je gaat vandaag tafels oefenen."
330 print:print
335 print"LET OP !"
340 for t=1 to 2000:next
345 :
400 rem hulp pagina
405 print chr$(147);
410 print"DIT MOET JE DOEN :"
415 print:print
420 print"Je geeft antwoord op de
    tafelsom in het"
425 print:print"venster. Als je je vergist,
    kun je het"
430 print:print"verbeteren met de spatie-balk."
435 print:print"Dat is die lange op
    het toetsenbord."
440 print:print
445 print"Als het antwoord is gegeven,
    moet je op"
450 print:print r$" drukken.":print
455 print"Als je deze pagina wilt zien, moet"
```

```
460 print:print"op de f 7-toets drukken."
465 print:print"VEEL SUCCES !!"
470 gosub 1500
475 :
500 rem schermopbouw
505 print chr$(147);
510 ti$="000000":rem klok op nul
515 print k1$;k2$;:for t=1to3:print k3$;:next
520 print k4$;k1$;k2$;k3$;k4$;k1$
525 print tab(10)k5$:for t=1to3
    :print tab(10)k6$:next
530 print tab(10)k7$:print tab(10)k5$
535 print tab(10)k6$:print tab(10)k7$
540 printk2$;k3$;k4$
545 :
600 rem scherm invullen
610 printleft$(ne$,3)tab(2)"Aan de beurt is:"
    tab(25)"Tafel:"
615 printleft$(ne$,5)tab(2)n$tab(25)tt
620 printleft$(ne$,9);
625 printtab(2)"Gedaan:"as;
630 printtab(15)"Goed:"g;
635 printtab(29)"Score:";spc(3-len(str$(sc)));sc;
640 printright$(ne$,5)
    :printtab(13)"                    "
645 printright$(ne$,2)
    :printtab(13)"                    "
650 printne$;chr$(29);"Druk als je klaar
    bent op "r$
655 :
700 rem tafelsommen
750 y=int(rnd(0)*10)+1:ify=yythen750
755 q=2:ify>9thenq=1
760 print left$(ne$,15)tab(10+q)y" x "tt" = "
770 print left$(ne$,15)tab(23+q);
775 goto 1000
780 :
800 rem controle
805 an=val(an$)
810 if an=y*tt then c$=" G O E D ! !"
    :g=g+1:goto 900
815 c$=" F O U T ! !"
820 :
900 rem tijdscontrole
905 h=0:if ti$>tyd$ then h=1
910 rem teller
915 as=as+1
920 rem score berekening
925 if as=0 then as=1
930 sc=int((g/as)*10)
935 yy=y
940 print left$(ne$,19)tab(14)c$
945 for t=1 to 1000:next
960 if h=1 then 2000
990 goto 600
995 :
1000 rem invoer van gegevens
1005 an$="":an=0
1010 get a$:if a$="" then 1010
1020 if asc(a$)= 13 then 800
1025 if asc(a$)=136 then 400
1030 if asc(a$)=137 then 100
1035 if asc(a$)=138 then 300
1040 if asc(a$)=139 then 5000
1045 if asc(a$)= 20 then 1010
1050 if asc(a$)=32 then print chr$(157)
     ;chr$(32)chr$(157);:goto 1080
1055 an$=an$+a$
1060 if len(an$)>3 then 800
1065 print a$;:goto 1010
1080 if len(an$)<1 then 1010
1085 an$=left$(an$,len(an$)-1)
1090 goto 1010
1100 :
1500 rem toets halen
1510 print:print"Druk op "r$" om
     verder te gaan."
1520 get a$:if a$="" then 1520
1530 if asc(a$) <> 13 then 1520
1540 return
1545 :
2000 rem einde oefening
2010 tt(x)=tt:n$(x)=n$:as(x)=as
     :g(x)=g:sc(x)=sc:td(x)=val(td$)
2020 print chr$(147);
2030 print"Dit was het einde van de oefening."
2040 print:print:print
2050 print"Je deed de tafel van "tt(x)
2060 print
2070 print"Je hebt van de "as(x)" opdrachten"
2080 print:print g(x)" goed gedaan !"
2090 print:print
2100 print"Je Score is: "sc(x)
2110 print:print"Druk op "r$"´en roep
     de volgende."
2120 gosub 1520:goto 300
2125 :
5000 a=0:x=x-1:rem administratie
5005 print chr$(147);
5010 print"NAAM"tab(15)"TAFEL  AANTAL GOED TYD"
5015 print k1$
5020 for aa=1 to 20
5025 a=a+1:if a>100 then a=1
5030 print n$(a);tab(15)tt(a)spc(4)as(a);
5035 print spc(4)g(a)spc(3)td(a)
5040 next aa
5045 get a$:if a$="" then 5045
5050 if asc(a$)=139 then a=0:goto 5000
5055 if asc(a$)=137 then 100
5060 if asc(a$)=134 then 300
5065 if asc(a$)=136 then 400
5070 if asc(a$)= 32 then 5005
5075 goto 5045
```

Tekstverwerkers als Vizawrite en Superscript worden vaak genoemd. De prijs van deze programma's ligt echter meestal boven de honderd (of zelfs tweehonderd) gulden. StarTexter 64 heeft door zijn mogelijkheden en prijs (circa *f* 175,-) in Duitsland al voor opschudding gezorgd. Reden genoeg dit programma van Sybex eens nader te bekijken.

Startexter 64

Budget-pakket, moeilijk, maar de moeite waard

Het beoordelen van een tekstprogramma is altijd moeilijk, vooral als men zelf gewend is aan een bepaald programma. Peter Boncz geeft toch een afgewogen mening over het budget-pakket van Sybex.

Startexter 64 start met een mooi beeld- en geluidseffect. Dat is niet strikt nodig, maar wel aangenaam. De indeling van het beeldscherm ziet er op het eerste gezicht ook al duidelijk en overzichtelijk uit.
Wat opvalt aan StarTexter, is dat veel al in het Nederlands werkt: boodschappen worden in onze taal weergegeven en veel afkortingen zijn van het Nedelandse woord afgeleid. Het programma werkt menugestuurd, maar voor directe commando's worden afkortingen gebruikt.
Er zijn in totaal drie hoofdmenu's:

- **F1** - het floppy menu. Via dit menu kan men laden saven etc. met de diskdrive.
- **F3** - het print- en formatteermenu. Met dit menu kan men de tekst bewerken of op de printer zetten.
- **F5** - het parametermenu. Hierin is het mogenlijk gegevens zoals de beeldschermkleur, het lettertype enz. vast te stellen.

Om direct opdrachten te geven is er een commandomodus, die men bereikt of verlaat d.m.v. de CTRL toets.

Edit

StarTexter 64 heeft de volgende mogelijkheden wat betreft het manipuleren met tekst:

☆ centreren van tekst (M van Midden)
☆ omschakelen van 40 naar 80 regels (S)
☆ Word-wrapping. (Dit is het overzetten van een woord naar de volgende regel als het woord te lang wordt voor de vorige regel)
☆ Zoeken en vervangen. (na het zoekwoord 'Z') Het is ook mogelijk om het zoekwoord te vervangen door een ander woord. Dit kan met of zonder bevestiging. (oud woord= nieuw woord of oud woord == nieuw woord)
☆ Plaatsen of weghalen van tabulatorstops. (runstop of shift/runstop)
☆ Blok operaties. A= begin van een blok, E = einde van een blok, L = naar links scrollen, R = naar rechts scrollen, T = verplaatsen, I = kopiëren, O = overschrijven en C = wissen.
☆ StarTexter kan samenwerken met de database Starfile. Zo is het bijvoorbeeld mogelijk standaardbrieven te schrijven.

Extra funkties

Dit is een indrukwekkende lijst opdrachten die in zijn totaliteit niet bij Vizawrite of Easyscript voorkomt, maar StarTexter 64 heeft zelfs nog meer, waarbij de overdaad in dit opzicht de gebruiksvriendelijkheid misschien zelfs een beetje schaadt:

⋆ rekenen en programmeren in BASIC.
⋆ De tekstverwerker beschikt over 6 verschillende lettersets die ook op de printer kunnen worden gezet.
⋆ Trema's, accenten en zelfgemaakte tekens (zoals bijvoorbeeld een telefoontje) kunnen worden gebruikt en op het scherm worden weergegeven.
⋆ De grafische tekens van de C-64 kunnen worden gebruikt.
⋆ Men kan met het bijgeleverde programma Starfont zelf tekens ontwerpen.
⋆ Teksten van Vizawrite, Textomat en Textomat Plus kunnen via het programma Converter worden omgezet naar StarTexter teksten.
⋆ De diskette waar StarTexter op staat is niet beveiligd, er kunnen dus ook kopieën voor eigen gebruik worden gemaakt. Door het ontbreken van beveiligingen is het programma een stuk sneller gebleven.
⋆ Overzichtsfunktie, door in de schrijfmodus op Commodore/shift te drukken wordt er een overzicht van het hele 'schrijfvel' gegeven.

Printen

De StarTexter tekstverwerker is Vizawrite en Easyscript op het gebied van printfaciliteiten de baas. Zelfs eenvoudige printers als de Commodore 803/801 (of compatibles) kunnen de 6 extra lettersets en de 10 zelf ontworpen tekens printen met de fontafdruk-funktie. De printer print dan precies wat er op het beeldscherm te zien is, hetgeen wel een tijdrovende aangelegenheid kan worden. Ook de betere parallel aangestuurde printers, die op de Userport worden aangesloten, kunnen worden aangestuurd en dan komen er naast de 6 fonts en de 10 zelfontworpen tekens ook nog eens de mogelijkheden bij van die printer zelf. dat kunnen dan lettertypen als Pica, Elite, Italic, Underline etc. zijn.

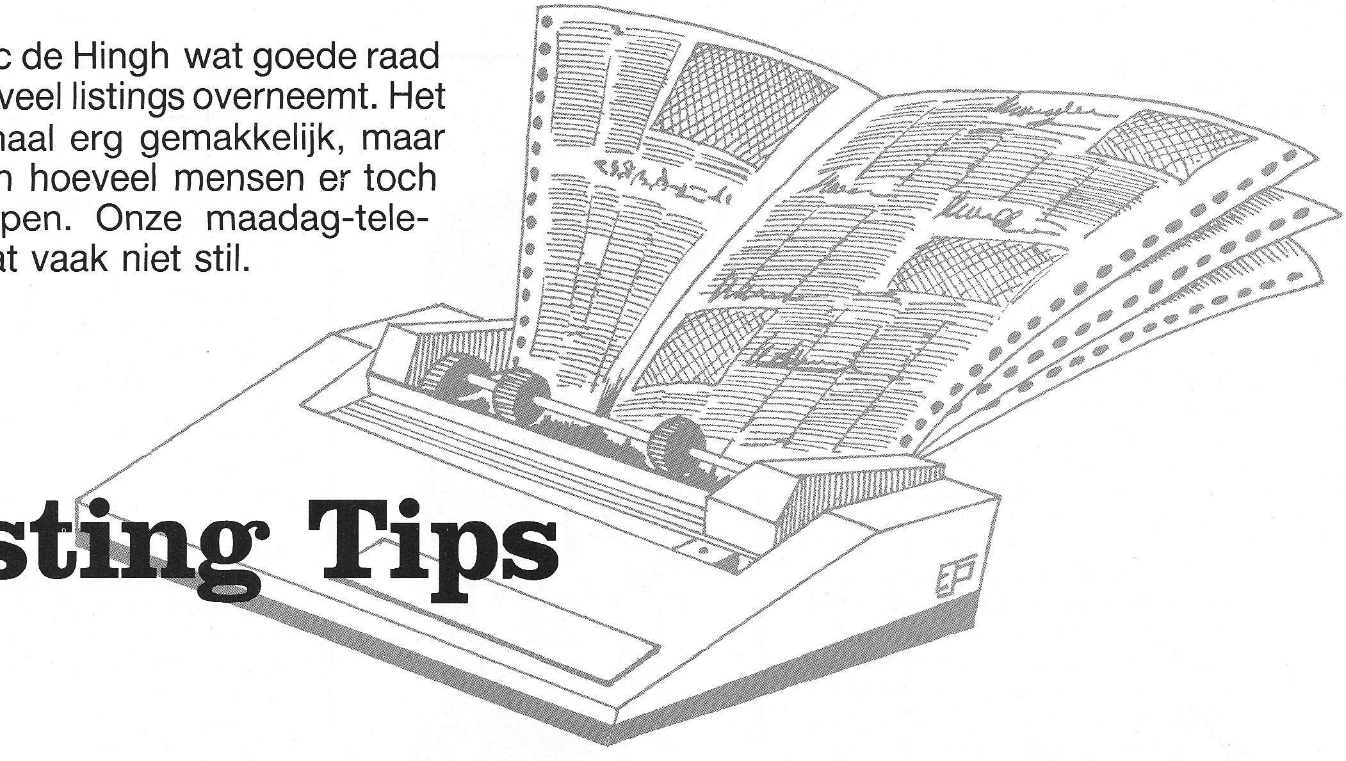
C-128 versie

Die is er helaas nog niet, en men moet zich haasten want het aantal tekstpakketten voor de 128 neemt snel toe. Er komt binnenkort dus een C-128 versie op de markt, met dezelfde (goede) eigenschappen. De belangrijkste verschillen met het C-64 programma zijn de 80 tekens op een regel en het grotere geheugen. ●

Conclusie

StarTexter 64 oogt goed en werkt zeer doeltreffend en degelijk. Toch zijn er wat nadelen. Deze nadelen liggen in de gebruiksvriendelijkheid voor de beginner. De startende tekstverwerker-gebruiker zal hier moeite hebben het overzicht te behouden, omdat de tekstverwerker zoveel functies heeft. De meer ervaren gebruiker zal zich op dit programma echter al gauw thuis voelen.

Van Marc de Hingh wat goede raad voor wie veel listings overneemt. Het lijkt allemaal erg gemakkelijk, maar we weten hoeveel mensen er toch in vastlopen. Onze maadag-telefoon staat vaak niet stil.



Listing Tips

Als computer-hobbyist kom je er niet onderuit: programma's overtypen uit computerbladen. Het is bijzonder leerzaam om te zien hoe mede-microbezitters programmeren en men kan eigen kennis verrijken met programmeertrucs van anderen. Voorwaarde is dan wel dat de afgedrukte listing goed wordt overgenomen en daar zit 'm nou net de grote moeilijkheid.

Vaak zit men uren te zweten en te ploeteren, om dan tot de ontdekking te komen dat de boel niet werkt, waarna men nog eens uren moet zoeken om de boel te controleren. En tenslotte werkt het programma dan, maar is de avond inmiddels om. Nergens voor nodig, want met wat goede wil en wat hulpmiddelen kan een listing overtypen een ontspannende en leerzame bezigheid worden. Gelukkig maakt uw favoriete Commodore-computerblad gebruik van checksum-controle getallen en dat is al de helft van het werk. Maar in dit artikel vindt u nog wat belangrijke tips.

Hoofdpijn

Een veel voorkomende klacht bij en na langdurig achter de computer zitten is hoofdpijn. Vaak komt dit door slechte verlichting, bijv. van het toetsenbord. Zorg altijd dat het toetsenbord en ook het blad waaruit u de listing overneemt, ongeveer even helder zijn als de monitor, dan hoeven uw ogen zich niet telkens in te stellen. Hetzelfde geldt voor de afstand tussen ogen en monitor/tv, toetsenbord en blad. Als deze afstanden (ongeveer) gelijk zijn werkt het veel prettiger.

Vind een goede plaats om het blad neer te leggen (voor, achter of naast toetsenbord?) om ook de dreigende nekpijn door het heen-en-weer kijken te vermijden. Hoofdpijn kan ook veroorzaakt worden door de kleur van het beeld of door gepiep of gebrom van de monitor/tv. Als u geen monochrome monitor heeft, zet dan tenminste het contrast wat lager en draai de kleur op zwart/wit of verander de schermkleuren. (Een donkere schermrand is rustgevend voor de ogen).

Even saven

Zo scheelt het al heel wat, maar als men te lang achter elkaar bezig blijft, komen de druppeltjes op het voorhoofd toch te voorschijn. Het is dan ook goed om geregeld even te pauzeren en dan is 'even saven' tussendoor belangrijk voor het geval er toevallig (?) een stroomstoring optreedt of u straks een foutje maakt en het met veel zorg ingetypte programma verliest. Met regelmatig een 'backupje' maken (professionals hebben er een aparte diskette voor) is helemaal opnieuw beginnen niet nodig. Denk er dan ook om, nooit het ingetypte te testen (runnen) vóór het saven. Als de computer dan vastloopt door een fout in het programma, is anders alles verloren.

Schermkleur

Omdat we u nu toch weer wat te doen willen geven, even een behulpzame listing. Onderstaand programma (C64) doet automatisch eens per half uur de schermkleur veranderen. De getallen tussen haakjes achter iedere regel zijn de checksumgetallen.

```
 50 REM Halfuurklok                       (215)
100 POKE 56580,255:POKE 56581,255 (9)
110 POKE 56590,17                         (52)
120 POKE 56582,181:POKE 56583,105 (5)
130 POKE 56591,81                         (54)
140 POKE 56589,127                       (110)
150 FORI=53200 TO 53210:READ X   (46)
160 POKE I,X:NEXT                         (32)
170 DATA 72,173,13,221,16,3,238   (191)
180 DATA 32,208,104,64                     (5)
190 POKE 792,208:POKE 793,207       (56)
200 POKE 56589,130                       (104)
210 PRINT"OK."                           (165)
```

Controle tekens

Probeer altijd te begrijpen wat u intypt. Als u alles letter voor letter overtypt, komen er veel meer fouten dan wanneer men bij het invoeren begrijpt wat er gebeurt. Analyseer eventueel het programma voor het overtypen, zodat de struktuur al een beetje bekend is.

In Commodore Info worden in listings bepaalde controletekens geschreven als woorden tussen spekhaken, die dan echter bij het overnemen van de listing weer vervangen moeten worden door de oorspronkelijke toets(en). Vooral de kleurcommando's zorgen hierbij voor moeilijkheden, omdat deze niet allemaal op het toetsenbord aangegeven worden. Voor de duidelijkheid nog eens een overzichtje:

ZWART	ctrl + 1 (BLK)
WIT	ctrl + 2 (WHT)
ROOD	ctrl + 3 (RED)
CYAAN	ctrl + 4 (CYN)
PURPER	ctrl + 5 (PUR)
GROEN	ctrl + 6 (GRN)
BLAUW	ctrl + 7 (BLU)

GEEL	ctrl + 8 (YEL)
Oranje	commodore + 1
Bruin	commodore + 2
L.Rood	commodore + 3
Grijs 1	commodore + 4
Grijs 2	commodore + 5
L.Groen	commodore + 6
L.Blauw	commodore + 7
Grijs 3	commodore + 8

Minder goed is het bij Commodore Info is gesteld met het afdrukken van graphics. Vaak is niet duidelijk welk teken er bedoeld wordt. Merk op dat de listings zijn afgedrukt in de tekstmode (kleine letters+hoofdletters). Hierdoor kunnen er tekens zijn die er op het scherm anders uitzien dan op papier. Zo geeft shift+aapje op het scherm een soort L in spiegelbeeld, terwijl het in de listing is afgedrukt als een worteltrekteken. Dit probleem wordt kleiner als u het scherm eveneens in de tekststand brengt met shift+commodore. Onnodig gezoek naar graphics tijdens het invoeren van een programma kan worden verholpen door van te voren de veel voorkomende tekens even op een kladblaadje te zetten met de bijbehorende toetsencombinatie.

Datafobie

Als u bij voorbaat besluit een listing niet over te nemen omdat de helft ervan in beslag wordt genomen door dataregels, dan lijdt u aan data-angst. Terecht, want interessant zijn die regels machinetaal- en spritedata niet echt. Hier zijn nog wat tips om het lijden te verzachten.

❶ Leg een papiertje onder de regel waar u bent zodat u niet na ieder getal moet zoeken waar u ook weer was. U kunt ook met een vinger aanwijzen waar u gebleven bent en met é é n hand typen. (Het zijn meestal toch alleen maar cijfers.)

❷ Probeer een vast ritme op te bouwen. Dat beperkt het aantal fouten sterk. Achtergrondmuziek kan hierbij helpen.

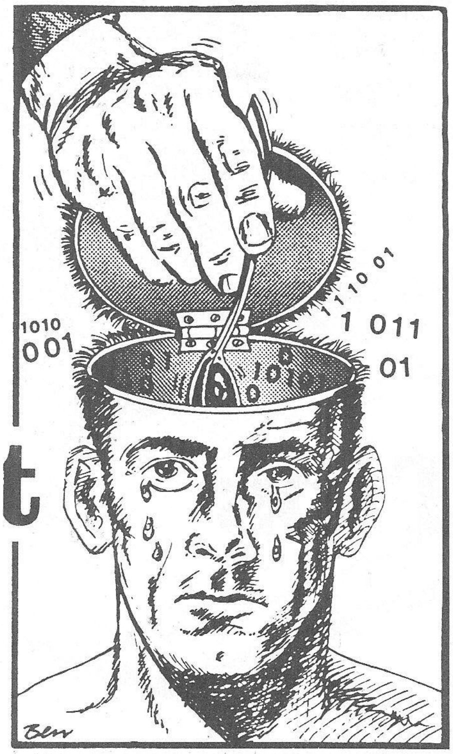
❸ Een veel gemaakte fout is een punt typen in plaats van een komma. Dat resulteert dan in een out of data error. Als u deze vergissing vaak maakt, plak dan een plakkertje op de komma-toets.

❹ Een tweede veel gemaakte fout: Als een dataregel zo is afgedrukt:

```
1000 DATA 12,204,0,192,2
     52,169,255
```

is men vaak geneigd een komma te typen achter de 2, terwijl er toch gewoon 252 staat. Tip: geef van te voren aan (met potlood) waar iets dergelijks het geval is.

❺ Roep de hulp in van een huisgenoot (nee, geen trouwe viervoeter) om de data voor te lezen. Dit kan wel ten koste van de snelheid gaan. (Ho! Wacht even! Wat was nou het eerste getal?). Ook eerst alle data



inspreken op een bandje behoort tot de mogelijkheden.

Checksum

Bij het gebruikte checksum-systeem worden de ascii-codes van alle karakters van een regel van het programma bij elkaar opgeteld. Zo ontstaat dan een getal tussen 0 en 255. Omdat 3+4 en 4+3 hetzelfde resultaat geeft, worden omdraaiingen van twee of meer tekens er niet uitgehaald. Typ daarom nooit te snel, want dan willen er nog wel eens omdraaiingen voorkomen en die komen er dus niet uit met de checsumtest. Spaties tellen niet mee voor het checksumprogramma. Let daarom ook goed op het aantal spaties. Teveel of te weinig spaties kunnen de schermopmaak bederven, hoewel de checksum geen fout aangeeft. Over het algemeen mogen spaties buiten aanhalingstekens echter wel naar believen weggelaten of toegevoegd worden omdat die verder geen funktie hebben. (Maar ze nemen wel geheugenruimte in.)

Let op de volgende uitzondering:

In gevallen als **100 Q=T AND(2 ↑ I)** moet wel degelijk de spatie tussen T en AND gehandhaafd blijven want anders ziet de commodore het als 100 Q=TAN D(2 ↑ I) met een syntax error als gevolg.

Zoals bekend hebben rem-regels (remark) geen enkele andere funktie dan uitleg over de werking van het programma. In principe hoeft u ze dus niet over te typen om het programma te laten werken. Maar u moet er dan wel voor oppassen dat er niet elders vanuit het programma naartoe wordt geGOTOd of geGOSUBd. Dan volgt namelijk een undef'd statement error.

Het veiligste is om bij een remregel een dubbele punt te typen om de betreffende regel 'bezet' te houden.

Dus 750 REM ---INITIALISATIE--- kan worden 750 : .

Dat scheelt een hoop overbodig typwerk.

Het verdient aanbeveling om in plaats van eerst het programma in te typen, te saven en vervolgens het checksumprogramma te laden, te runnen, het ingetypte programma opnieuw te laden en te testen op checksumgetallen, van te voren al "syntax checksum" te laden en te runnen, zodat men ook tussendoor de checksum van de tot dan toe ingevoerde regels kan toetsen. Het is tenslotten een stukje machinetaal, dat in het geheugen aanwezig blijft. Het is ook veel prettiger regelmatig de checksum te controleren dan e'e'n keer op het eind.

Voor de liefhebbers geeft het volgend programma (C-64) een alternatief voor de gebruikelijke syntax checksum. Met deze utility verschijnt onmiddellijk na iedere ingevoerde regel de bijbehorende checksum op het scherm. Het werkt inderdaad heel geruststellend om direct te weten dat de regel goed ingetypt is. Een tip: schrijf de checksumgetallen in een afwijkende kleur achter elke regel van het programma. Dan hoeft u niet telkens heen en weer te kijken tussen programma en checksumlijst. Hier volgt het programma (de getallen tussen haakjes na iedere regel zijn de checksumgetallen).

```
 50 REM GEEFT CHECKSUM                  (77)
 60 REM NA IEDERE REGEL                 (59)
100 C=0:FOR I=53000 TO 53125:READX
    (144)
110 POKE I,X:C=C+X:NEXT                (148)
120 IF C=15914 THEN 150                (193)
130 PRINT"DATAFOUT!"                    (86)
140 PRINT"TEST DE CHECKSUM!":END
    (212)
150 SYS58451:POKE770,8:POKE771,207
    (173)
160 PRINT"OK."                         (165)
200 DATA 160,0,177,209,201,35,240,11
    (171)
210 DATA 201,32,208,90,200,177,209,196
    (30)
220 DATA 213,208,245,32,19,166,144,78
    (246)
230 DATA 32,204,255,173,134,2,72,169,1
    (25)
240 DATA 141,134,2,169,35,32,210,255
    (179)
250 DATA 166,20,165,21,32,205,189,169
    (243)
260 DATA 58,32,210,255,160,4,169,0,133
    (22)
270 DATA 2,177,95,240,12,201,32,240,5
    (222)
280 DATA 24,101,2,133,2,200,208,240,
    169 (56)
```

```
290 DATA 0,166,2,32,205,189,104,141,
    134 (67)
300 DATA 2,169,32,32,210,255,165,211
    (178)
310 DATA 197,213,208,245,169,13,32,210
    (31)
320 DATA 255,169,255,133,21,32,96,165
    (249)
330 DATA 134,122,132,123,32,115,0,201
    (207)
340 DATA 35,240,242,32,121,0,76,141,
    164 (64)
```

Als het controlegetal van een regel niet klopt, maar er is echt geen fout te vinden in die regel, moet u letten op het volgende:
Is nergens een spatie gebruikt in plaats van een shift-spatie of andersom?
Moet ik misschien een ander grafisch teken gebruiken of moet dat minteken geen shift-sterretje zijn?
Is die 1 geen L of 0 geen O?
Als dit allemaal niet het geval is kan het een 'misser' zijn. In het volgende nummer vindt u dan een oplossing en anders biedt de listingservice telefoonlijn misschien uitkomst.

Foutje

Als alles dan is ingetypt, checkgesumd, verbeterd en gesaved kan het programma uiteindelijk worden getest. Meestal zal het dan goed werken, maar er kunnen toch nog een paar dingen mis gaan. Als voorbeeld noem ik een insteek- of uitbreidingsmodule, die roet in het eten kan gooien bij een gebruikte machinetaalroutine. De oplossing is om voor het runnen een quit of kill commando te geven.
Heel lastig is het feit, dat oudere versies van de C-64 soms een bepaald programma toch niet aan kunnen, dat is een oudere bug, die overigens software-matig te elimineren is.
Verder kan er ondanks de checksumtest toch nog een foutje zijn blijven zitten. Dit zal dan meestal een omdraaiing zijn. Als er een foutmelding wordt gegeven kan men met wat kennis van Basic betrekkelijk gemakkelijk de fout opsporen. Daarbij blijkt de regel, waar de foutmelding naar wijst, niet altijd de regel waarin de fout zit. Een typisch voorbeeld is de syntax error in een dataregel, die men krijgt, wanneer geprobeerd wordt een string uit data in een numeriek variabele te lezen m.b.v. read.
Het is ook mogelijk dat tijdens de uitvoer van het programma opeens het scherm leeg wordt met alleen 'ready'. (zoals na stop-restore). Dit wil zeggen dat er iets fout is met een machinetaalroutine. De fout moet gezocht worden in de dataset.

Als er geen foutmelding gegeven wordt, maar er gebeurt gewoon niet, wat er zou moeten gebeuren zit er niets anders op dan heel het programma nog eens door te kijken en te vergelijken met de in het blad afgedrukte listing. Met een printer kan de foutieve versie uitgeprint worden om te vergelijken.

Na een regel die precies een schermregel lang is, niet vergeten op return te drukken. De cursor staat al wel op het begin van de volgende schermregel, maar het ingetypte is nog niet verwerkt. De volgende programmaregel wordt dan (foutief) aan de vorige vastgelijmd.
De computer verwerkt nooit meer dan twee regels (80 karakters) tekst per regel. Als een ingetypte regel toch langer lijkt, wil dat zeggen dat de programmeur afkortingen heeft gebruikt, zoals '?' voor print en p shift-o voor poke. Dat scheelt dan wat tekentjes. Verder kosten spaties (buiten aanhalingstekens) ook ruimte, die gewonnen kan worden.
Gebruik geen shift-return i.p.v. return om de regel af te sluiten. Dan wordt het ingevoerde namelijk genegeerd.

Veel plezier met de listings,

Marc de Hingh

In dit deel van onze cursus machinetaal behandelen we de manier waarop informatie in de computer wordt geordend. Daarna zullen we zien hoe deze elektrische informatie kan worden vertaald naar voor de mens bruikbare gegevens. We zullen dan te maken krijgen met een belangrijk computertalstelsel: het tweetallige stelsel. Omdat die stap van elektrische informatie naar getallen een erg belangrijke is zullen we dit artikel er helemaal aan besteden.

CURSUS MACHINETAAL

door Sjoerd Bakker

Deel 2: Bits, bytes en het tweetallige stelsel

Het gaat dit keer om de manier waarop de computer intern aan het rekenen slaat en hoe we op de eenvoudigste manier zo met cijfers kunnen werken, dat de computer die begrijpt. Om de logische lijn in deze cursus aan te geven, zullen we om te beginnen het voorafgaande deel nog kort samenvatten.

In deel 1 zagen we dat een microprocessor niets anders is dan een hele grote verzameling open/dichtschakelaars. Met het schema van een lampjesschakeling lieten we zien hoe de processor met behulp van zes schakelaars de getallen 1-3 met 1-3 zou kunnen vermenigvuldigen. Daarna hadden we het over de transistor: de "computerschakelaar" die elektrisch open en dicht wordt gezet. De opdrachten voor het open en dicht zetten van de schakelaars, dus aan de microprocessor, zijn opgeslagen in het geheugen van de computer. Dat geheugen bestaat uit ruim 65.000 zgn. geheugenplaatsen die elk een eigen nummer, oftewel adres, hebben.

Zo'n geheugenplaats kan als inhoud (niet te verwarren met het adres) een waarde van 0-255 hebben.

We zagen dat de waarden beschouwd kunnen worden als opdracht-codes voor de processor, maar ook als data-informatie bij een opdracht. Tenslotte gaven we met een voorbeeldprogramma aan hoe een aantal opeenvolgende opdrachten in de praktijk wordt uitgevoerd.

Sturing schakelaars

De oplettende lezer zal in het bovenstaande verhaal een tegenstrijdigheid hebben ontdekt. Want: enerzijds zouden de geheugenplaatsen open/dicht-informatie voor de ingangsschakelaars van de processor moeten bevatten, maar anderzijds omschrijven we de inhoud ervan als "een waarde van 0-255". Hoe vallen deze verschillende benaderingen voor een en dezelfde toestand met elkaar te rijmen? Daarvoor moeten we om te beginnen weer even terug naar het "spanningsniveau" van de computer.

ADRES	INHOUD
0	0-255
	0-255
	0-255
	0-255
	0-255
	0-255
	0-255
65535	0-255

ELKE GEHEUGENPLAATS HEEFT EEN NUMMER, OFTEWEL EEN ADRES. EEN GEHEUGENPLAATS BEVAT ALTIJD EEN WAARDE VAN 0-255 DIE DE INHOUD WORDT GENOEMD. DEZE WAARDES KUNNEN OP VERSCHILEENDE MANIEREN WORDEN BESCHOUWD. BIJ PROGRAMMEREN IN MACHINETAAL VORMEN ZE VAAK OPDRACHT-CODES VOOR DE MICROPROCESSOR.

Laag en hoog

In een vorig artikel zagen we dat een computer maar twee niveau's van elektrische spanning kent. Het zijn het "lage" 0-Volts niveau en het "hoge" 5-Volts niveau. Als er een schakelaar in de computer open staat zal de stroom zich niet kunnen bewegen van het "hoge" naar het "lage" spanningsniveau. Zetten we de schakelaar echter dicht, dan zal er 5 Volt op een lijn komen te staan en vloeit er stroom naar de aarde (0 Volt). Het open en dicht zetten van de schakelaars gaat ook met behulp van deze twee spanningsniveau's. Afhankelijk van de soort schakelaar maak je hem open door het "lage" of het "hoge" niveau op de "schakelknop" te zetten. Het andere niveau zorgt er dan voor dat de schakelaar dicht gaat. Over hoe het open en dicht zetten van de schakelaars precies in zijn werk gaat zullen we het verder niet hebben. Zoals eerder gezegd komen we dan op het gebied van de elektronika en dat onderwerp valt buiten het kader van dit blad.

Bits en bytes

Het is dus zo, dat de machinetaalopdrachten zich in de vorm van "lage" en "hoge" spanning in het geheugen van de computer bevinden. Met andere woorden: het geheugen van de computer bestaat uit (zeer veel) kleine gebiedjes waarvan de elektrische spanning of "laag" of "hoog" is. Zo'n gebiedje noemen we een **bit.** Een bit is de *kleinste informatieeenheid binnen* de computer, en je kunt je hem het beste voorstellen als een micro

```
BITNUMMER:                 7 6 5 4 3 2 1 0

ELEKTRISCHE
TOESTAND:                  L H H L H L L H
```

EEN BYTE WORDT GEVORMD DOOR 8 BITS. BITS ZIJN ZEER KLEINE GEBIEDJES, WAARVAN DE ELEKTRISCHE SPANNING 'LAAG OF 'HOOG' IS. DE BITS VAN EEN AFZONDERLIJKE BYTE HEBBEN ELK EEN EIGEN BIT-NUMMER. DE MEESTE BYTES ZIJN TEVENS GEHEUGENPLAATSEN, MAAR HET HOEFT NIET.

scopisch dun en kort elektriciteitsdraadje waar dan die "lage" of "hoge" spanning op staat. Als we echter zouden moeten werken met geheugenplaatsen die uit een enkele bit bestaan, zouden we niet veel kunnen beginnen. We zouden de processor dan maar twee verschillende opdrachten kunnen laten uitvoeren. Het "lage" spanningsniveau symboliseert dan de ene opdracht en het "hoge" niveau de andere. Daarom bestaat een geheugenplaats niet uit een enkele, maar uit 8 bij elkaar horende bits. Een dergelijk groepje van 8 noemen we ook wel een Byte. Een geheugenplaats bestaat dus uit 8 gebiedjes waarvan de elektrische spanning "laag" of "hoog" is.
Als we werken met de inhoud van een geheugenplaats houden we ons altijd bezig met de toestanden van die 8 bits tegelijk. Omdat we echter wel van elk afzonderlijk bit van een geheugenplaats de toestand willen kunnen weten, hebben ze ieder een eigen nummer. Die nummers lopen niet van 1-8, zoals je zou verwachten, maar van 0-7. Als we een byte tekenen zetten we de bits ervan meestal op een horizontale rij. De volgorde van links naar rechts wordt dan: bit 7, 6, 5, 4, 3, 2, 1, 0.

De elektrische toestanden van alle bits samen kunnen een bepaalde opdracht vormen voor de microprocessor.

In de tekening hebben we in de afzonderlijke bits het elektrische spanningsniveau ervan aangegeven. De niveau's van bits 7-0 zijn dus respektievelijk: laag, hoog, hoog, laag, hoog, laag, laag en hoog. Als de processor deze combinatie van spanningsniveau's op de "schakelknoppen" van zijn 8 ingangsschakelaars krijgt, zal hij de opdracht "optellen" gaan uitvoeren. Op dezelfde manier kunnen andere "laag"/ "hoog"combinaties weer de uitvoering van andere opdrachten tot gevolg hebben. Hoeveel combinaties er in totaal mogelijk zijn zullen we zo zien.

We kunnen dus zeggen dat het geheugen van de computer uit ruim 65.000 bytes bestaat, die elk een nummer van 0-ruim 65.000 hebben.

Dit nummer is het adres van de geheugenplaats, en het dient om al die verschillende geheugenplaatsen uit elkaar te kunnen houden. Een enkele geheugenplaats bestaat weer uit 8 verschillende bits, een byte, die genummerd zijn van 0-7. De elektrische spanning van een bit is altijd "laag" (0 Volt) of "hoog" (5 Volt). De toestanden van alle bits samen vor men de inhoud van de geheugen plaats. En over die inhoud gaan we het nu in het bijzonder hebben.

```
GEWICHT:   100  10   1
CIJFER :     8   7   6
             |   |   |
             |   |   L---6*  1=  6
             |   |
             |   L-------7* 10= 70
             |
             L-----------8*100=800 +
                              876
```

HET TIENTALLIGE STELSEL WERKT MET DE CIJFERS 0-9 EN EEN ONBEPERKT AANTAL CIJFERPLAATSEN. ELKE CIJFERPLAATS HEEFT EEN EIGEN 'GEWICHT'. VAN RECHTS NAAR LINKS ZIJN DIE GEWICHTEN: 1, 10, 10*10, 10*10*10, 10*10*10*10 ETC. OM ONS DE GROOTTE VAN EEN TIENTALLIG GETAL VOOR TE KUNNEN STELLEN, MOETEN WE DE AFZONDERLIJKE CIJFERS MET HET BIJBEHORENDE GEWICHT VERMENIGVULDIGEN EN VERVOLGENS AL DEZE UITKOMSTEN BIJ ELKAAR OPTELLEN.

Spanningen en getallen

In principe is er niets op tegen om de inhoud van een geheugenplaats aan te duiden met: bit 7 is "laag", bit 6 is "hoog", bit 5 is "hoog" etc. We bekijken de computer dan vanaf het elektriciteitsniveau, waarop hij in feite funktioneert. Zo'n twintig jaar geleden was het heel normaal dat programma's werden geschreven door het omzetten van schakelaars, waarmee je dan de bits van een geheugenplaats met de hand een voor een "laag" of "hoog" maakte. De "laag"/ "hoog"manier van voorstellen heeft echter twee nadelen. Ten eerste hebben we dus de zojuist genoemde omslachtigheid. Het zou natuurlijk veel handiger zijn als je de inhoud van een geheugenplaats korter kon omschrijven, en je dus alle bits in een keer kon veranderen.

Rekenmethode

Het tweede nadeel is van veel ernstiger aard. Zoals het er nu uitziet zijn we niet in staat om met de inhouden van geheugenplaatsen te rekenen. Een inhoud die "laag", "hoog", "hoog", etc. is, valt natuurlijk geen getal te noemen. Hoe zouden we in vredesnaam twee van die inhouden bij elkaar op moeten tellen? En stel dat het ons lukt: wat moeten we aan met een resultaat van 8 "laag"/ "hoog"waardes? Welke getallen hebben we dan bij elkaar opgeteld, en wat is het resultaat? Om met dit laatste probleem te beginnen: om te kunnen rekenen met de inhouden van geheugenplaatsen blijkt het noodzakelijk te zijn om die inhouden op een bepaalde manier overeen te laten komen met getallen. Dit kunnen we het beste doen door alle mogelijke "laag"/"hoog"combinaties van de 8 bits systematisch te rangschikken. De eerste combinatie in de rij laten we dan het getal 0 voorstellen; de tweede combinatie wordt het getal 1, en zo gaat het door tot we alle mogelijke "laag"/"hoog"combinaties hebben gehad. Het voordeel van zo'n systematische rangschikking is, dat we de elektrische toestanden heel gemakkelijk kunnen "vertalen" naar getallen en omgekeerd. Maar welk systeem moeten we toepassen bij het ordenen van de mogelijke combinaties? Om hier achter te komen moeten we eerst het talstelsel dat we dagelijks gebruiken eens nader gaan bekijken.

Het tientallige stelsel

Voor we verder gaan spreken we af dat "geschreven getallen" in het vervolg bepaalde aantallen voorstellen, en dat cijfers die aantallen in verschillende talstelsels kunnen symboliseren.
Een voorbeeld: het getal honderd kan de hoeveelheid honderd appels aanduiden. In het tientallige stelsel wordt deze hoeveelheid cijfermatig uitgedrukt door het getal 100. Het tientallige, ofwel decimale talstelsel waar wij mee bekend zijn kent tien getallen om hoeveelheden mee voor te stellen. Met andere woorden: de cijfers 0-9 symboliseren de aantallen nul-negen. Verder kent het stelsel een onbeperkt aantal cijferplaatsen, die elk een bepaald gewicht vertegenwoordigen.
Een voorbeeld: het getal 876 kan een aantal appels symboliseren. Om ons dit aantal voor te kunnen stellen moeten we 6 keer een enkele appel voor ons zien, met

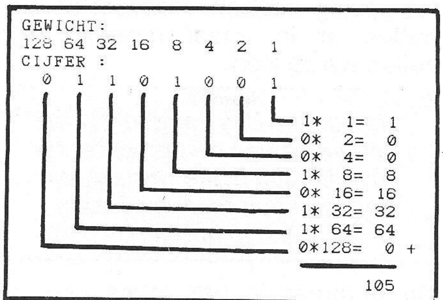
daarnaast 7 keer 10 appels en daar weer naast 8 keer 100 appels. De waardes 1, 10, en 100 vormen dan de gewichten van de achtereenvolgende cijferplaatsen. Het cijfer op een bepaalde cijferplaats geeft aan met hoeveel het gewicht moet worden vermenigvuldigd.

Werking

De werking van het stelsel is heel simpel. De aantallen nul-negen worden weergegeven met de cijfers 0-9. Dan hebben we alle beschikbare cijfers gehad en roepen we voor de hoeveelheid tien de hulp in van een tweede cijferplaats, waar we een 1 in zetten. De rechter cijferplaats "springt weer op 0". Vervolgens laten we de linker 1 staan en beginnen we rechts weer te tellen van 1-9 voor de hoeveelheden elf-negentien. Dan springt de linker cijferplaats van 1 naar 2, en zo gaat het door tot we bij de negen-en-negentig (!) komen. Nu zijn ook twee cijferplaatsen niet genoeg om het volgende aantal, honderd, weer te geven en nemen we er links een derde bij waar we een 1 neerzetten. De andere twee "springen" weer op 0". Zoals je ziet is het principe heel eenvoudig, maar het is nuttig om er even bij stil te staan. Wij zijn zo zeer aan ons eigen talstelsel gewend dat we ons er nauwelijks meer bewust van zijn dat het werkt zoals hierboven beschreven. Daardoor kunnen we ons ook nauwelijks voorstellen dat het net zo goed mogelijk is om hoeveelheden weer te geven in een ander stelsel dan het tientallige.

Het tweetallige stelsel

Het is eigenlijk een heel kleine stap van de "laag"/"hoog"combinaties die de 8 bits van een byte vormen naar het tweetallige stelsel. Immers, een bit is een informatie-eenheid waarmee een onderscheid valt te maken tussen twee verschillende toestanden. Het enige verschil met een cijferplaats in het tientallige stelsel is dat daarmee een onderscheid valt te maken tussen tien verschillende toestanden. Ook zijn "laag" en "hoog" natuurlijk geen cijfers. maar daar valt wel wat op te vinden. Het is in de positieve computerlogica de gewoonte om de "lage" spanning van een bit voor te stellen met het cijfer 0,



HET TWEETALLIGE STELSEL WERKT MET DE CIJFERS 0-1 EN EEN ONBEPERKT AANTAL CIJFERPLAATSEN. (EEN BYTE HEEFT ER ECHTER 8). ELKE CIJFERPLAATS HEEFT EEN EIGEN GEWICHT. VAN RECHTS NAAR LINKS ZIJN DIE GEWICHTEN: 1, 2, 2*2, 2*2*2, 2*2*2*2 ETC. OM ONS DE GROOTTE VAN EEN TWEETALLIG GETAL VOOR TE KUNNEN STELLEN MOETEN WE DE GEWICHTEN VAN DE CIJFERPLAATSEN WAAR EEN 1 STAAT BIJ ELKAAR OPTELLEN.

en de "hoge" spanning met het cijfer 1. Ziedaar de twee cijfers 0 en 1, waarmee we een tweetallig, oftewel **binair**, talstelsel kunnen gaan vormen. En omdat een byte uit 8 bits bestaat hebben we dus de beschikking over **8** cijferplaatsen. Omdat de bits van een byte altijd een bepaalde elektrische spanning hebben zullen we, in tegenstelling tot het tientallige stelsel, altijd van alle 8 tweetallige cijferplaatsen het cijfer vermelden; ook als we links een aantal opeenvolgende 0-waardes hebben. Het getal 1001 schrijven we dus bijvoorbeeld als 00001001. Laten we nu eens gaan kijken hoe het tweetallige stelsel werkt. Met het cijfer 00000000 geven we natuurlijk de hoeveelheid nul weer; dat spreekt voor zich. Ook de hoeveelheid een geeft weinig problemen. Deze wordt aangeduid met het getal 00000001. Nu zijn we door al onze cijfers heen en moeten we er dus een cijferplaats bij nemen. Net als in het tientallige stelsel maken we de eerstvolgende cijferplaats 1, en de rechtse "springt weer op 0". De hoeveelheid twee geven we dan weer met het getal 00000010. De volgende hoeveelheid, drie, krijgen we door de meest rechtse cijferplaats met 1 te verhogen, zodat drie wordt voorgesteld door 00000011. Nu zitten de tot dusver gebruikte twee cijferplaatsen alweer vol, en moeten we er opnieuw een bij nemen. Vier schrijven we dan als 00000100. Voor de hoeveelheid vijf beginnen we weer rechts met tellen en we krijgen 00000101. Zes wordt vervolgens 00000110 en zeven 00000111. We zitten nu weer vol, zodat de hoeveelheid acht wordt voorgesteld met 00001000.

Bijgaand overzicht geeft voor de hoeveelheden nul-vijftien de tweetallige en vervolgens de tientallige schrijfwijze. De vier meest rechtse bits zitten dan vol. Kijk er maar eens rustig naar, en let ook op de gewichten van de cijferplaatsen die in het tientallige stelsel worden voorgesteld. In het tientallige stelsel zelf waren die gewichten van rechts naar links:

1,10,10*10,10*10*10,10*10*10*10
en in het tweetallige stelsel zijn ze:
1, 2, 2*2, 2*2*2, 2*2*2*2 etc.

We zien verder dat je met een halve byte (4 bits) 16 verschillende hoeveelheden weer kunt geven. Het totaal aantal mogelijke combinaties voor een hele byte is dus gelijk aan (16*16=) 256. Vandaar dat het mogelijk is om de inhoud van een geheugenplaats zowel in termen van "lage" en hoge" spanningen aan te duiden, als met een tweetallig (00000000-11111111) of een tientallig (0-255) getal. In het volgende deel van deze cursus zullen we zien dat de inhoud van een geheugenplaats meestal wordt aangeduid in nog een ander talstelsel: het *zestientallige* of hexadecimale.

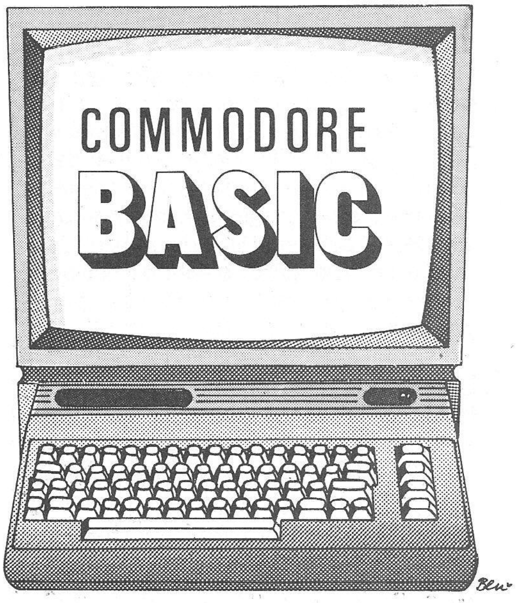
HOEVEELHEID	TWEETALLIG								TIENTALLIG	
	128	64	32	16	8	4	2	1	10	1
NUL	0	0	0	0	0	0	0	0		0
EEN	0	0	0	0	0	0	0	1		1
TWEE	0	0	0	0	0	0	1	0		2
DRIE	0	0	0	0	0	0	1	1		3
VIER	0	0	0	0	0	1	0	0		4
VIJF	0	0	0	0	0	1	0	1		5
ZES	0	0	0	0	0	1	1	0		6
ZEVEN	0	0	0	0	0	1	1	1		7
ACHT	0	0	0	0	1	0	0	0		8
NEGEN	0	0	0	0	1	0	0	1		9
TIEN	0	0	0	0	1	0	1	0	1	0
ELF	0	0	0	0	1	0	1	1	1	1
TWAALF	0	0	0	0	1	1	0	0	1	2
DERTIEN	0	0	0	0	1	1	0	1	1	3
VEERTIEN	0	0	0	0	1	1	1	0	1	4
VIJFTIEN	0	0	0	0	1	1	1	1	1	5

DEZE TABEL GEEFT DE HOEVEELHEDEN NUL-VIJFTIEN WEER IN HET TWEETALLIGE EN HET TIENTALLIGE STELSEL

Aan het eigenlijke programmeren zijn we nog niet of nauwelijks toegekomen. We vinden het echter beter om je eerst de noodzakelijke basisinformatie te geven, dan om meteen met flitsende listings te gaan gooien waarvan je toch niet weet hoe ze werken. Er heeft in Commodore Info tenslotte al een eerdere cursus gestaan, die wat dieper inging op het programmeren in machinetaal. ■

De beginnende programmeur kan met hulp van deze serie artikelen leren hoe leuk het is om zelf méér te doen met de Commodore, dan alleen het spelen van voorgekookte spelletjes en het gebruiken van andere gekochte software-applicaties. Stap voor stap worden in deze cursus door Jan Bodzinga de elementaire feiten van Basic behandeld. De stof is daarbij bewust eenvoudig gehouden, zodat ook de volslagen 'leek' op dit gebied erdoor kan uitgroeien tot een redelijk programmeur. Maar ook de meer gevorderde 'Basiccer' zal ongetwijfeld nog wel dingen tegenkomen, waarmee zijn kennis kan worden aangevuld.

Basis Basic



Deel 10

Sorteren & Arrays (II)

In het vorige artikel zijn we begonnen met een heel nieuw type Basic-variabele waar we het nog niet eerder over hebben gehad, de ARRAY. We hebben daarbij kennis gemaakt met de mogelijkheden die array-variabelen bieden, om veel - min of meer gelijke gegevens - op een korte en snelle manier in een programma te verwerken. Daarbij gaat het niet alleen om numerieke gegevens, zoals functies, formules en getallen, maar we kunnen met hulp van een *stringarray* ook alfanumerieke gegevens op die manier verwerken. En vooral deze laatste optie geeft ons, samen met de computer, de mogelijkheid om namen, adressen en andere (persoons)gegevens, kortom alle alfanumerieke gegevens op een bijna professionele manier te verwerken. Maar eerst zullen we moeten weten hoe we te werk moeten gaan om reeksen getallen naar grootte te rangschikken.

De variabele-soort 'ARRAY' is op zich niet zo ingewikkeld. Als we ervan uitgaan, dat ieder onderdeel van de array zijn eigen, specifieke waarde kan krijgen in een programma, kunnen we iedere index een aparte variabele-naam toekennen, waardoor we weliswaar een veel langer, maar misschien beter te begrijpen, programma hebben. In aflevering 9 zijn we van het omgekeerde uitgegaan. Om de zaak nog even op te halen, drukken we hier een deel van de listing af van de vorige keer, waarbij de 'array' als variabele zijn intrede deed:

```
10 REM GEMIDDELDE 2
20 REM Berekenen gemiddelde van 7
   getallen
30 REM GEBRUIK ARRAY A()
50 I=0 : REM TELLER
60 DIM A(7) : REM ARRAY DEFINITIE
70 T1 =0 : REM TOTAAL
80 RES=0 : REM RESULTAAT
100 REM BEGIN PROGRAMMA
110 FOR I = 1 TO 7
120 READ A(I)
130 T1 = T1 + A(I)
140 NEXT I
210 RES = T1 / 7
220 PRINT "GEMIDDELDE ="; RES
230 END
300 DATA 445,14,54,6633,234,66,1465
```

Zoals iedereen kan zien, al is het maar aan de REM-regels, wordt er in dit programma een gemiddelde berekend van 7 getallen. De getallen worden stuk voor stuk ingelezen in een variabele die hoort bij de array A(). De **index** van deze numerieke array wordt daarbij aangegeven door de waarde van I uit de loop van regel 110 tot 140. Daarnaast wordt ook het totaalaantal opgeteld in variabele T1. Dit laatste is mede reden van herplaatsing van deze listing, want tot mijn schande moet ik bekennen, dat ik me in les 9 door het Commodore Basic dialect te pakken heb laten nemen. In de vorige aflevering stond de variabele TOT in plaats van T1. Terwijl ik al zo vaak iedereen ervoor heb gewaarschuwd, dat in namen van variabelen bij Commodore nooit onderdelen van Basic-opdrachten mogen zitten, heb ik me er zelf in laten lopen, doordat in TOT het Basicwoord TO voorkomt. Het probleem is, dat in de meeste andere Basic-vormen deze naamgeving volkomen legaal is. Desondanks sorry voor de fout.

De array-waarden

Hoewel in bovenstaande listing het gemiddelde (RES) perfect berekend wordt, is de ARRAY A() in feite nog niet nodig. We zouden voor het lezen (READ) een gewone numerieke variabele kunnen gebruiken, die dan bij T1 wordt opgeteld en meteen weer wordt gebruikt voor het volgende getal in de loop. Maar als we dat doen, zijn we wel meteen de afzonderlijke waarden van de 7 getallen kwijt en daar willen we nog wat mee doen, om de uitkomst te kunnen controleren.

Als eerste voegen we de volgende regels aan de listing toe:

```
230 PRINT : PRINT "WAARDEN : "
240 FOR I = 0 to 7
250 PRINT I, A(I)
260 NEXT
299 END
```

Als we nu het programma weer runnen, zullen we niet alleen de uitkomst (RES) op het scherm zien, maar daarnaast ook de afzonderlijke waarden van de zeven getallen.

Het ziet er dan ongeveer zo uit op het scherm:

```
GEMIDDELDE = 1273
WAARDEN :
0         0
1         445
2         14
3         54
```

4	6633
5	234
6	66
7	1465

We zien hier zowel de waarde van de loopvariabele, met daarachter de waarde van A(I), de array met index I. Hieraan is duidelijk te zien, dat de getallenreeks uit de DATA van regel 300 netjes vanaf A(1) is ingelezen, tot het laatste getal dat via READ in A(7) is terecht gekomen. De array-variabele A(0) heeft als waarde 0, omdat hij niet is meegenomen bij de READ-opdracht.
We zien dus, dat de waarden van de getallen nog steeds aanwezig is in het computergeheugen. Dat betekent, dat we ook nog van alles kunnen doen met deze waarden. Een voor de hand liggende mogelijkheid is het rangschikken op grootte van deze 7 getallen. Dit sorteren heeft een aparte en belangrijke plaats bij het programmeren, omdat hiermee de computer pas efficient gebruikt gaat worden.

Sorteren

Het rangschikken of sorteren van getallen en alfanumerieke gegevens is één van de taken waarvoor de computer een prima hulpmiddel is. Bij ons voorbeeld van 7 getallen zullen we de sortering waarschijnlijk net zo snel handmatig kunnen uitvoeren, anders wordt het echter als we een groot aantal getallen willen gaan rangschikken. Voor het sorteerprincipe maakt het echter niets uit, hoeveel getallen er moeten worden gerangschikt, mits er op zijn minst twee getallen zijn.
Er bestaat een scala van diverse **algoritmes** om te sorteren. En om eerlijk te zijn is Basic nu niet bepaald de meest voor de hand liggende computertaal om mee te gaan sorteren, maar wel de taal die de sorteerprincipes het eenvoudigst kan voorstellen. De eenvoudigste (en langzaamste) methode van sorteren die we hier zullen behandelen is de **Verwissel** methode. In het engels heet dit de 'Exchange-sort'. Deze manier is gemakkelijk te programmeren en nog goed te begrijpen. Daarnaast kennen we bv. nog de 'Bubble-sort', de 'Tournament-sort', 'Shell-Metzner' en de 'binaire methode'. Uiteindelijk komt het er bij alle manieren op neer, dat de gegevens netjes worden gerangschikt van laag naar hoog of omgekeerd. Het ligt aan het geduld van de programmeur en de gebruiker welke methode de voorkeur verdient, terwijl ook de wijze waarop de gegevens in de computer zitten belangrijk is bij de keuze van de methode.

Verwisselen

Om het sorteren te kunnen illustreren gaan we weer uit van de basis van ons vorige programma. We nemen de getallen en het inlezen van gegevens met kleine wijzigingen over:

```
10 REM VERWISSEL-SORT
20 REM Rangschikken van 7 getallen
30 REM MET HULP VAN ARRAY A()
40 I=0 : REM LUSTELLER
50 J=0 : REM LUSTELLER
60 DIM A(7) : REM ARRAY DEFINITIE
100 REM BEGIN PROGRAMMA
110 FOR I = 1 TO 7
120 READ A(I)
130 NEXT I
500 DATA 445,14,54,6633,234,66,1465
590 END
```

We kunnen dit stuk programma als basis gebruiken voor alle sorteermethodes. Voor de algoritme van de **Verwissel-methode** voegen we de volgende regels aan het programma toe:

```
200 REM SORTEERROUTINE
210 FOR I = 1 TO 6
220 FOR J = I+1 TO 7
230 IF A(I) > A(J) THEN GOSUB 600
240 NEXT J
250 NEXT I
........
600 REM VERWISSELEN
610 TP=0: REM OPSLAG
620 TP=A(I)
630 A(I)=A(J)
640 A(J)=TP
650 RETURN
```

De routine om het sorteren uit te voeren vinden we in de regels 200 tot 250. Daar zien we, dat deze methode gebruik maakt van een **dubbele loop** met I en J als variabelen. Als we goed kijken zien we, dat sorteren in feite niets anders is, dan het vergelijken van twee getallen uit de reeks. Is getal 1 kleiner (of gelijk) aan getal twee, dan gebeurt er niets en blijven ze op hun eigen plaats in de array staan. Is het eerste getal groter dan het tweede, dan wordt via de subroutine uit regel 600-650 ervoor gezorgd dat de waarde van de twee getallen van plaats in de array wordt verwisseld.

Structuur

De gevorderde programmeur zal direct zien, dat deze manier van wisselen in een subroutine het geheel nog meer vertraagt, zeker omdat de opdrachten van 600 tot 640 zondermeer passen na de IF..THEN test van regel 230. Voor de duidelijkheid is de structuur van dit programma echter met een subroutine opgezet.
Bekijk eerst de verwisselroutine eens, omdat daarin de echte truc schuilt van het sorteren. Andere dialecten kennen Basic-opdrachten als SWAP A(I),A(J) etcetera, waarbij hetzelfde gebeurt, maar voor de Commodore moeten we als programmeur het wisselen zelf schrijven. Bij entree in de subroutine hebben we twee waarden in de array A() met respectievelijk index I en J. Deze beide moeten worden omgedraaid, waarna het programma via RETURN wordt vervolgd. Dit omdraaien gebeurt op de enig mogelijke (omslachtige) manier. Stel dat we in A(I) een waarde hebben van 8 terwijl in A(J) de waarde 3 zit.
Laten we het wisselen volgen met deze twee getallen. Om dit goed te laten lopen hebben we een extra variabele nodig, die als tijdelijke opslag fungeert. Het stuivertje wisselen gaat dan ook zo:

regel 610 : TP=0 : A(I)=8 : A(J)=3
regel 620 : TP=8 : A(I)=8 : A(J)=3
regel 630 : TP=8 : A(I)=3 : A(J)=3
regel 640 : TP=8 : A(I)=3 : A(J)=8

Je ziet dat er zondermeer een extra variabele nodig is, om het verwisselen van waarden tussen twee variabelen te kunnen laten gebeuren.

Sorteerlussen

Om het rangschikken te bewerken maakt deze methode gebruik van twee Basic-lussen (loops). Het principe komt erop neer, dat er telkens in de eerste lus (I) een waarde wordt genomen, die vervolgens in de tweede lus (J) wordt vergeleken met alle waarden van de array die een hogere index hebben. Vandaar ook de afwijkende start en eindwaarden van de lusvariabelen. Om het proces in zijn geheel te kunnen volgen zullen we het sorteerprogramma wat moeten uitbreiden. Daarbij zullen we ook de officiële syntax van de **verwisselsort** in het programma zetten in plaats van de regels 200 - 250. Daardoor is het programma universeel en kan voor het sorteren van N-waarden worden gebruikt.

```
10 REM SORTEERROUTINE
20 REM Rangschikken van N getallen
30 REM MET HULP VAN ARRAY A()
40 I=0 : REM LUSTELLER
50 J=0 : REM LUSTELLER
60 N=7 : RAM AANTAL TE SORTEREN
70 DIM A(N) : REM ARRAY DEFINITIE
100 REM BEGIN PROGRAMMA
110 FOR I = 1 TO N
120 READ A(I)
130 NEXT I
200 REM VERWISSEL-SORT
210 FOR I = 1 TO N-1
220 FOR J = I+1 TO N
230 IF A(I) > A(J) THEN GOSUB 600
240 NEXT J
250 NEXT I
500 DATA 445,14,54,6633,234,66,1465
599 END
600 REM VERWISSELEN
610 TP=0: REM OPSLAG
```

```
615 GOSUB 700
620 TP=A(I)
630 A(I)=A(J)
640 A(J)=TP
645 GOSUB 700
650 RETURN
700 REM STATE OF SORT
710 PRINT "I=","I," J=",J,
720 FOR K = 1 TO N : PRINT A(K);: NEXT
730 PRINT
740 RETURN
```

Er is nogal wat veranderd aan de listing. De sorteerroutine heeft nu de variabele N als aantal, waarbij ook duidelijk wordt, dat lus I loopt tot het voorlaatste getal in de reeks (N-1), terwijl lus J steeds begint bij de waarde die één hoger ligt dan de array-waarde die op dat in lus I aan de beurt is (I+1).

Om op het scherm te kunnen volgen wat er gebeurt tijdens het doorlopen van de lussen hebben we een aparte routine geschreven (700-740) die de **status** weergeeft van alle getallen in de array A(). Deze routine wordt steeds vlak voor en na iedere wissel aangeroepen, waarbij de waarden van de lusvariabelen I en J worden afgedrukt om het verloop te illustreren. Tijdens het runnen krijgen we het volgende:

I= 1	J= 2	445	14	54	6633	234	66	1465
I= 1	J= 2	14	445	54	6633	234	66	1465
I= 2	J= 3	14	445	54	6633	234	66	1465
I= 2	J= 3	14	54	445	6633	234	66	1465
I= 3	J= 5	14	54	445	6633	234	66	1465
I= 3	J= 5	14	54	234	6633	445	66	1465
I= 3	J= 6	14	54	234	6633	445	66	1465
I= 3	J= 6	14	54	66	6633	445	234	1465
I= 4	J= 5	14	54	66	6633	445	234	1465
I= 4	J= 5	14	54	66	445	6633	234	1465
I= 4	J= 6	14	54	66	445	6633	234	1465
I= 4	J= 6	14	54	66	234	6633	445	1465
I= 5	J= 6	14	54	66	234	6633	445	1465
I= 5	J= 6	14	54	66	234	445	6633	1465
I= 6	J= 7	14	54	66	234	445	6633	1465
I= 6	J= 7	14	54	66	234	445	1465	6633

Aan de waarden van I kunnen we nu precies aflezen waar en hoe de wissels tijdens het sorteren hebben plaatsgevonden. Wat direct opvalt is, dat er in alle gevallen tijdens het doorlopen van de buitenste lus (I) minstens 1 keer een wissel plaats vindt. Zou er daar niet gewisseld hoeven worden, omdat het bijvoorbeeld een gesorteerde reeks getallen betreft, dan toch zal het sorteren even lang duren. In een paar gevallen komen er in dit voorbeeld zelfs meer dan één wissel voor.

Wisselen

De eerste wissel vinden we als I=1 en J=2. De te vergelijken variabelen zijn dan A(1) en A(2) met inhoud 445 en 14. Dit klopt uiteraard, want 445 is groter dan 14 daarom vindt de wissel plaats. De rest van deze loop gebeurt er niets meer, omdat nu 14 in A(I) is gekomen en 14 is het kleinste getal uit de reeks.

Bij de tweede ronde wordt 445 gewisseld met 54. De derde gang wijkt af van de vorige, omdat er daar meer wissels plaatsvinden. De lus begint met in A(I) de waarde 445. Deze wordt eerst vergeleken met A(4) waarbij niet wordt gewisseld. A(5) heeft als inhoud 234 waardoor de subroutine 600-650 wordt uitgevoerd. Na de RETURN en zit nu dus 234 in A(3) terwijl A(5) de waarde 445 heeft gekregen. Als de J-loop opnieuw wordt doorlopen wordt een vergelijking gemaakt tussen A(3) = 234 en A(6) = 66. Omdat A(6) kleiner is dan A(3) krijgen we hier opnieuw een wissel. Het eigenaardige van deze sorteermethode is, dat er naast het rangschikken van de kleinste getallen, tijdens de verwerking ook andere getallen op een onverwachte manier van plaats veranderen. In deze serie is dat goed te zien, want bij de start stond het getal 234 op de vijfde plaats in de reeks, na een paar wissels is het getal naar achteren geschoven, terwijl uiteindelijk de vierde plaats in de reeks de juiste is. De sortering wordt op dezelfde manier door het programma afgemaakt.

Exchange-sort

De hiervoor beschreven en geïllustreerde **verwissel- methode** om te sorteren heeft naast het feit dat er weinig regels nodig zijn om de routine in een Basic-listing op te nemen, niet zo veel voordelen. Ten eerste is de methode nogal omslachtig, waardoor hij voor Basic niet goed te gebruiken is. Daar komt bij, dat er ook bij het opnieuw rangschikken van een gesorteerde reeks data met deze methode, nagenoeg evenveel tijd door de computer wordt gebruikt.

Een voordeel is het gemak waarmee deze routine voor 'snelle' programmeurs te gebruiken is.

Bubble-sort

Een betere, gestroomlijnder sorteermethode is de veel toegepaste **Bubble-sort**. Bubble betekent belletje en dat is ook wat er hier gebeurt. Per stuk worden de getallen vergeleken met het direkt daaropvolgende getal. We maken in Basic nog wel gebruik van een dubbele lus, maar onze tabel zal laten zien, dat er op een snellere manier door de lussen wordt gelopen. Bovendien wordt bij de Bubble-sort gebruikt gemaakt van een vlag, waardoor de routine onmiddelijk stopt en niet, zoals de exchange-sort zondermeer blijft doorgaan tot beide lussen zijn afgewerkt.
Om de bubble-sort in het programma te verwerken dienen de reges 200 - 250 en 600 - 640 te worden aangepast op de volgende manier:

```
200 REM BUBBLE-SORT
210 FOR I = N-1 TO 1 STEP -1: VLAG = 0
220 FOR J = 1 TO I
230 IF A(J) > A(J+1) THEN VLAG = 1 :
    GOSUB 600
240 NEXT J
250 IF VLAG = 0 THEN I = 1
260 NEXT I
. . . . . . . . . . . . .
600 REM VERWISSELEN
610 TP=0: REM OPSLAG
615 GOSUB 700
620 TP=A(J)
630 A(J)=A(J+1)
640 A(J+1)=TP
650 GOSUB 700
660 RETURN
```

Als het programma is herschreven en we runnen het voor de eerste keer, dan krijgen we de volgende tabel op het scherm:

I= 6	J= 1	445	14	54	6633	234	66	1465
I= 6	J= 1	14	445	54	6633	234	66	1465
I= 6	J= 2	14	445	54	6633	234	66	1465
I= 6	J= 2	14	54	445	6633	234	66	1465
I= 6	J= 4	14	54	445	6633	234	66	1465
I= 6	J= 4	14	54	445	234	6633	66	1465
I= 6	J= 5	14	54	445	234	6633	66	1465
I= 6	J= 5	14	54	445	234	66	6633	1465
I= 6	J= 6	14	54	445	234	66	6633	1465
I= 6	J= 6	14	54	445	234	66	1465	6633
I= 5	J= 3	14	54	445	234	66	1465	6633
I= 5	J= 3	14	54	234	445	66	1465	6633
I= 5	J= 4	14	54	234	445	66	1465	6633
I= 5	J= 4	14	54	234	66	445	1465	6633
I= 4	J= 3	14	54	234	66	445	1465	6633
I= 4	J= 3	14	54	66	234	445	1465	6633
I= 0	J= 4	14	54	66	234	445	1465	6633

weggezet. De Bubble methode zal dan ook sneller gaan, naarmate de originele reeks beter op volgorde ligt. Een compleet We zien hier dat de getallen uit de reeks **uitsluitend** verschuiven naar de plaats waar ze horen, en niet zoals bij de verwisselsort maar links of rechts even worden gesorteerde reeks getallen, of strings, zal bij de Bubble-methode dan ook nagenoeg geen uitvoertijd meer kosten.

Return

Tot zover de uitvoerige toelichting op het rangschikken van eenvoudige getallen in Basic. Het kan beslist geen kwaad de voorbeelden grondig te bekijken, want deze twee methodes geven een schets van de mogelijkheden die er zelfs in Basic zijn, om met de Commodore professioneel aan het werk te gaan. De volgende keer zullen we deze en andere methodes om gegevens naar grootte te rangschikken toepassen op alfanumerieke data. ●

Jan Bodzinga

Print-out

Met o.a. Ufo Attack – Op z'n kop – Blackjack – Checksum C-16 – Tempo-typen C-16.

Op dit moment dat ik dit stukje aan het schrijven ben is het donker en vochtig weer buiten. Het is een genot om in een verwarmde kamer achter de vertrouwde Commodore te zitten. Ik verwacht daarom ook dat er de komende weken een berg listings wordt ingezonden. Maar nu we het toch daarover hebben denkt U wel om de spelregels, dus vermeld: naam en adres zo volledig mogelijk, het type machine waarvoor het bedoeld is en een gebruiksaanwijzing van het programma.

Mochten er nog problemen zijn dan belt U met ons informatienummer maar dit kan alleen 's maandagavonds tussen 17.00 en 21.00 uur. Tel. 02155-25162. Als U er niet direkt doorkomt probeer het dan na een uur of acht dan is het wat rustiger.
Veel succes met het intypen van deze listings. ●

Inhoud van dit listingdeel

Syntax Checksum

Het overtikken van een listing kan een heel karwei zijn en als u een beetje normaal mens bent dan maakt u daarin beslist een aantal fouten. Nu is niets moeilijker om de fouten uit je eigen werk te halen. Al geruime tijd geleden heeft Jan Bodzinga hiervoor een zgn. **Checksum-programma** geschreven. Om de vele nieuwe lezers van Commodore-Info te helpen volgt hieronder nog een keer een volledige uitlegh over de werking van dit programma, waarmee het, hoe vreemd dat misschien ook lijkt, echt mogelijk is om met behulp van dit programma de fouten in elke door ons geplaatste listing op te sporen.
Hiervoor gaat u als volgt te werk:

❶ U tikt de listing heel zorgvuldig over en SAVEt hem voordat u het programma RUNt op een diskette of een cassette.

❷ U tikt het RUN commando in. Mocht het programma de boodschap **'Fout in dataregels!'** geven dan heeft u een fout bij het overtikken gemaakt. Herstel de fout en SAVE de verbeterde versie. Mocht het programma met de boodschap **'data is weggezet checksum testten met sys ....'** komen dan is tot dusver alles goed. Het programma is nu in een stukje machinetaalgeheugen gezet. Als u het NEW commando geeft blijft het toch in de computer staan.

Alle door ons geplaatste programma's zijn in Basic geschreven. Als u een programma heeft overgetikt SAVE het eerst, mocht er iets mis gaan dan hoeft niet de gehele listing opnieuw te gaan intikken. Als u nu een programma op fouten wilt gaan controleren dan kunt u dat in het geheugen laden (wel eerst het checksumprogramma hebben gerund). Vervolgens typt u zonder het programma te runnen de opdracht **sys 49152** (C-64) of **sys 1536** (C-16 en Plus/4) in.

Als alles goed is gegaan loopt er nu een rij regelnummers over het scherm met getallen erachter. Dezelfde lijst staat ook achter elk door ons geplaatste programma. Wijkt nu een nummer achter een regelnummer af van het nummer dat in het blad staat dan heeft u in die regel iets anders ingetikt dan er in het blad stond. U kunt de stroom getallen d.m.v. de RUN/STOP toets pauzeren en weer vervolgen met de F1 of F7 toets. Het is uitermate belangrijk dat u goed met dit programma overweg kunt en mocht U het niet goed werkend krijgen bel dan gerust even met onze listingservice telefoonlijn.

```
  1 rem ********************************
    **
  2 rem basic loader "SYNTAX.CHECKSUM"
  3 rem na de commando's 'run' en 'new'
  4 rem blijft dit programma in het ge-
  5 rem heugen. laad het te testen pro-
  6 rem gramma en tik daarna sys 49152.
  7 rem ********************************
    **
 10 i=49152  :rem beginadres
 20 reada:ifa<0then40:rem data ingeleze
    n
 30 pokei,a:i=i+1:b=b+a:goto20
 40 if b<>16844thenprint"[CLR-HOME]fout
     in dataregels!":b=0:end
 50 poke49184,148:poke49185,192
 55 i=49300
 60 read a: ifa<0then80
 70 pokei,a:b=a+b:i=i+1:goto60
 80 if b<>20068thenprint"[CLR-HOME]fout
     in dataregels! (vanaf regel 240)":
    b=0:end
 90 print"data is weggezet"
 95 print"checksum testen met sys49152"
100 data 165,43,166,44,133,163,134,164,
     169, 147
110 data 32,210,255,160,0,240,3,32,73,
     192
120 data 32,73,192,208,1,96,32,225,255,
      208
130 data 3,76,116,164,32,81,192,32,73,1
    92
140 data 240,12,201,32,240,247,24,101,1
    67,133
150 data 167,76,37,192,166,167,169,0,13
    2,168
160 data 32,205,189,169,13,32,210,255,1
    64, 168
170 data 76,17,192,200,208,2,230,164,17
    7,163
180 data 96,162,0,189,123,192,240,6,32,
    210
190 data 255,232,208,245,32,73,192,170,
    32,73
200 data 192,132,168,32,205,189,162,3,1
    69,32
210 data 32,210,255,202,208,250,169,0,1
    33,167
220 data 164,168,96,82,69,71,69,76,32,0
230 data -1
240 data 165,197,201,3,240,7,201,4,240
250 data 6,76,148,192,76,34,192,169
260 data 147,32,210,255,76,161,192
270 data -1

*** EINDE LISTING ***

syntaxchecksum listtestprogramma

regel 1    249
regel 2    84
regel 3    125
regel 4    2
regel 5    246
regel 6    152
regel 7    249
regel 10   157
regel 20   64
regel 30   38
regel 40   57
regel 50   14
regel 55   251
regel 60   192
regel 70   42
regel 80   244
regel 90   245
regel 95   237
regel 100  183
regel 110  158
regel 120  232
regel 130  183
regel 140  96
regel 150  96
regel 160  127
regel 170  71
regel 180  223
regel 190  73
regel 200  79
regel 210  109
regel 220  106
regel 230  225
regel 240  16
regel 250  163
regel 260  92
regel 270  225

ready.
```

HANGMAN

Dit is weer een computerversie van het overbekende spel galgje. Als U alleen wil spelen op de lange winteravonden laat de computer U een woord raden.

```
 99 rem ****initialisering**********
    ***
100 dim w$(999):b=0:poke53280,0:poke
    53281,0:poke646,1
109 rem ****intro*******************
    **
110 print"{CLR-HOME}{neer}"
120 print"{2xspatie}{RVS-aan} {RVS-uit}
    {2xspatie}{RVS-aan} {RVS-uit}{2xspat
    ie}{RVS-aan}{2xspatie}{RVS-uit}{2xsp
    atie}{RVS-aan} {RVS-uit}{2xspatie}
    {RVS-aan} {RVS-uit} {RVS-aan}{4xspat
    ie}{RVS-uit} {RVS-aan} {RVS-uit}
    {3xspatie}{RVS-aan} {RVS-uit}{2xspat
    ie}{RVS-aan}{2xspatie}{RVS-uit}{2xsp
    atie}{RVS-aan} {RVS-uit}{2xspatie}
    {RVS-aan} {RVS-uit}"
130 print"{2xspatie}{RVS-aan} {RVS-uit}
    {2xspatie}{RVS-aan} {RVS-uit} {RVS-a
    an} {RVS-uit}{2xspatie}{RVS-aan}
     {RVS-uit} {RVS-aan}{2xspatie}{RVS-u
    it} {RVS-aan} {RVS-uit} {RVS-aan}
     {RVS-uit}{4xspatie}{RVS-aan}{2xspat
    ie}{RVS-uit} {RVS-aan}{2xspatie}
    {RVS-uit} {RVS-aan} {RVS-uit}{2xspat
    ie}{RVS-aan} {RVS-uit} {RVS-aan}
    {2xspatie}{RVS-uit} {RVS-aan} {RVS-u
    it}"
140 print"{2xspatie}{RVS-aan}{4xspatie}
    {RVS-uit} {RVS-aan}{4xspatie}{RVS-ui
    t} {RVS-aan} {RVS-uit} {RVS-aan}
    {2xspatie}{RVS-uit} {RVS-aan} {RVS-u
    it} {RVS-aan}{2xspatie}{RVS-uit}
     {RVS-aan} {RVS-uit} {RVS-aan} {RVS-
    uit} {RVS-aan} {RVS-uit} {RVS-aan}
    {4xspatie}{RVS-uit} {RVS-aan} {RVS-u
    it} {RVS-aan}{2xspatie}{RVS-uit}"

150 print"{2xspatie}{RVS-aan} {RVS-uit}
    {2xspatie}{RVS-aan} {RVS-uit} {RVS-a
    an} {RVS-uit}{2xspatie}{RVS-aan}
     {RVS-uit} {RVS-aan} {RVS-uit}{2xspa
    tie}{RVS-aan} {RVS-uit} {RVS-aan}
     {RVS-uit}{2xspatie}{RVS-aan} {RVS-u
    it} {RVS-aan} {RVS-uit}{3xspatie}
    {RVS-aan} {RVS-uit} {RVS-aan} {RVS-u
    it}{2xspatie}{RVS-aan} {RVS-uit}
     {RVS-aan} {RVS-uit}{2xspatie}{RVS-a
    an} {RVS-uit}"
160 print"{2xspatie}{RVS-aan} {RVS-uit}
    {2xspatie}{RVS-aan} {RVS-uit} {RVS-a
    an} {RVS-uit}{2xspatie}{RVS-aan}
     {RVS-uit} {RVS-aan} {RVS-uit}{2xspa
    tie}{RVS-aan} {RVS-uit} {RVS-aan}
    {4xspatie}{RVS-uit} {RVS-aan} {RVS-u
    it}{3xspatie}{RVS-aan} {RVS-uit}
     {RVS-aan} {RVS-uit}{2xspatie}{RVS-a
    an} {RVS-uit} {RVS-aan} {RVS-uit}
    {2xspatie}{RVS-aan} {RVS-uit}"
170 print"{2xspatie}""""""""""""""""""""""
    """"""""""""""""""""""""""
180 printtab(11)"{neer}(c) h. broekm
    an"
190 print"{2xneer}{3xspatie}dit is d
    e computerversie van het "
200 print"{2xspatie}alom bekende spe
    l ophangertje."
210 print"{2xspatie}de een typt een
    woord in en de ander"
220 print"{2xspatie}probeert dit woo
    rd te raden."
230 print"{2xspatie}of de computer n
    eemt een woord en u"
240 print"{7xspatie}moet dit probere
    n te raden."
250 print"{3xspatie}als je een zin n
    eemt zet dan het "
260 print"{6xspatie}'/' teken tussen
     de woorden "
270 print"{neer}"
280 print"{neer}{11xspatie}{RVS-aan}
    sla een toets aan"
290 geta$:ifa$=""then290
300 input"{CLR-HOME}{2xneer}{3xrechts}
    1 of 2 spelers ";sp
310 if sp<1 or sp>2 then 300
320 if sp=2 then 360
330 b=b+1:readw$(b):if w$(b)="stop"then
     350
340 goto 330
350 u=int(rnd(1)*b):wo$=w$(u):goto37
    0
360 print"{CLR-HOME}{neer}{rechts}ge
    ef uw woord/zin (niet zichtbaar)"
    :poke646,0:input wo$:poke646,1
369 rem ****zet woord op scherm ****
    ***
370 print"{CLR-HOME}":le=len(wo$):dim
    ar$(le),bl$(le):b=-1
380 forx=1tole:ar$(x)=mid$(wo$,x,1):
    next
390 forx=1tole:ifar$(x)="/"thenbl$(x
    )=ar$(x):next
400 bl$(x)="-":next:gosub470
410 c=0:geta$:ifa$=""then410
420 y=y+1
430 forx=1tole:ifa$=ar$(x)thenbl$(x)
    =a$:c=1
440 nextx:ifc=1thengosub470
450 ifc=0thengosub480
460 goto410
470 print"{HOME}"tab(222):forx=1tole
    :printbl$(x);:nextx:gosub 730:return
479 rem ****teken galgje************
    **
480 b=b+1:ifb=0thenforx=1to11:poke18
    65+x,120:nextx:return
490 ifb=1thenforx=0to560step40:poke1
    271+x,97:nextx:return
500 ifb=2thenforx=1to6:poke1230+x,12
    1:next:return
510 ifb=3then:poke1274,78:poke1313,7
    8:poke1352,78:return
520 ifb=4thenforx=0to120step40:poke1
```

```
      276+x,93:next:return
 530 ifb=5thenpoke1435,85:poke1436,67
     :poke1437,73:poke1475,74:poke147
     6,67:poke1477,75
 540 ifb=5thenreturn
 550 ifb=6thenpoke1516,93:poke1555,10
     2:poke1556,102:poke1557,102:poke
     1595,102
 560 ifb=6thenpoke1596,102:poke1597,1
     02:poke1635,102:poke1636,102:poke
     1637,102
 570 ifb=6thenreturn
 580 ifb=7thenpoke1554,77:poke1513,77
     :return
 590 ifb=8thenpoke1558,78:poke1519,78
     :return
 600 ifb=9thenpoke1675,78:poke1714,78
     :poke1753,120:return
 610 ifb=10thenpoke1677,77:poke1718,7
     7:poke1759,120
 619 rem ****resultaat***************
     ***
 620 print"{HOME}{15xneer}"tab(20)"sc
     haam je het was"
 630 printtab(20)"{neer}{RVS-aan}";wo
     $
 640 printtab(20)"{HOME}{21xneer}"tab(
     20)"nog een keer j/n"
 650 geta$:ifa$=""then650
 660 ifa$="j"thenb=0:run
 670 ifa$<>"n"then640
 680 printtab(25)"{neer}{RVS-aan}tot
     ziens!!":end
 690 print"{HOME}{15xneer}"tab(20)"he
     t was "
 700 printtab(20)"{RVS-aan}";wo$
 710 printtab(20)"{neer}geraden in "
 720 printtab(20)y" beurten":goto640
 729 rem *****check*****************
     *
 730 fort=1to le:if bl$(t)="-"then return
 740 nextt:goto690
 741 rem ****************************
     **
 742 rem ** zet in deze data's je
     **
 743 rem ** eigen woorden (max. 999)
     **
 744 rem ** zet altijd! als laatste
     **
 745 rem ** woord 'stop'.
     **
 746 rem ****************************
     **
 750 data"floppy","sfinx","huisdeur"
 760 data"cassette","computer","hond"
 770 data"horloge","tentoonstelling"
 780 data"joystick","tijdschrift"
 790 data"vloerbedekking","sleutel"
 800 data"quartz","socialisme","imper
     ial"
9999 data"stop"

*** EINDE LISTING ***
```

ôREGEL 99	116	REGEL 480	165
REGEL 100	136	REGEL 490	116
REGEL 109	53	REGEL 500	203
REGEL 110	129	REGEL 510	198
REGEL 120	233	REGEL 520	24
REGEL 130	49	REGEL 530	119
REGEL 140	141	REGEL 540	233
REGEL 150	213	REGEL 550	223
REGEL 160	49	REGEL 560	13
REGEL 170	3	REGEL 570	234
REGEL 180	117	REGEL 580	90
REGEL 190	8	REGEL 590	103
REGEL 200	141	REGEL 600	197
REGEL 210	15	REGEL 610	47
REGEL 220	221	REGEL 619	224
REGEL 230	46	REGEL 620	105
REGEL 240	129	REGEL 630	51
REGEL 250	53	REGEL 640	159
REGEL 260	90	REGEL 650	104
REGEL 270	238	REGEL 660	191
REGEL 280	23	REGEL 670	39
REGEL 290	104	REGEL 680	175
REGEL 300	201	REGEL 690	45
REGEL 310	130	REGEL 700	34
REGEL 320	82	REGEL 710	169
REGEL 330	46	REGEL 720	214
REGEL 340	31	REGEL 729	221
REGEL 350	108	REGEL 730	97
REGEL 360	73	REGEL 740	56
REGEL 369	60	REGEL 741	123
REGEL 370	238	REGEL 742	12
REGEL 380	191	REGEL 743	205
REGEL 390	11	REGEL 744	241
REGEL 400	156	REGEL 745	132
REGEL 410	193	REGEL 746	123
REGEL 420	63	REGEL 750	114
REGEL 430	166	REGEL 760	155
REGEL 440	148	REGEL 770	208
REGEL 450	128	REGEL 780	229
REGEL 460	30	REGEL 790	97
REGEL 470	35	REGEL 800	202
REGEL 479	164	REGEL 9999	13

UFO ATTACK

Een eenvoudig spelletje gemaakt door Henk Broekman.
De bedoeling is dat men de UFO's door middel van de spatiebalk neerschiet. Met de z en de x toets is de tank te verplaatsen. Met de spatiebalk kan je schieten.

```
100 rem *******ufo attack***********
110 print"{CLR-HOME}{WIT}":sc=0:ti$=
    "000000"
120 poke53281,0:poke53280,0
130 fort=0to39:poke1984+t,102:next
140 rem ******poke ufo*************
150 print"{HOME}{RVS-aan}score= "sc:
    y=int(rnd(2)*20)
160 s=1024+(y*40):a=a+1:b=s+a:pokeb,
    107:pokeb+1,87:pokeb+2,115:pokeb
    -1,32
170 ifa>=37thenpokeb,32:pokeb+1,32:poke
    b+2,32:a=0:goto150
```

```
180 ifm=1thenreturn
190 rem *****besturing************
200 geta$
210 ifa$="z"thenx=x-1:ifx<=0thenx=0
220 ifa$="x"thenx=x+1:ifx>=39thenx=3
    9
230 ifa$=chr$(32)then300
240 print"{HOME}{21xneer}"tab(x)"{3xspat
    ie}] "
250 printtab(x)" ${RVS-aan}{3xspatie}
    {RVS-uit}$ "
260 printtab(x)" WWWWW "
270 ifti$>="000060"then410
280 goto160
290 rem ******vuur kogel af********
300 m=1:fort=1824+x+3 to 1024+x+3 step
    -40:poket,66:poket+40,32:gosub16
    0
310 p=peek(t-40):ifp=107orp=87orp=11
    5thenpoket,32:goto350
320 next
330 poke1024+x+3,32:m=0:goto160
340 rem ******geluid***************
350 pokeb,1 :pokeb+1,8 :pokeb+2,8
360 v=54272:forl=0to24:pokev+1,0:next
    :pokev+5,11:pokev+6,16:pokev+24,
    15
370 pokev+4,129:pokev+1,10:pokev,111
    :fort=1to300:next
380 forw=15to0step-1:pokev+24,w:next

390 pokeb,32:pokeb+1,32:pokeb+2,32:m
    =0:a=0:poket-40,32:sc=sc+1:goto1
    50
400 rem *****resultaat************
410 print"{CLR-HOME}{RVS-aan}resulta
    at"
420 print"{neer}je hebt er ";sc;" ge
    raakt in 60 sec."
430 ifsc>uxthenux=sc
440 print"{neer}uw score{5xspatie}= "
    ;sc
450 print"{neer}hoogste score= ";ux
460 print"{3xneer}{RVS-aan}hit any k
    ey"
470 geta$:ifa$=""then 470
480 goto 110

*** EINDE LISTING ***
```

ôREGEL 100	37	REGEL 250	237
REGEL 110	56	REGEL 260	52
REGEL 120	37	REGEL 270	85
REGEL 130	231	REGEL 280	32
REGEL 140	198	REGEL 290	38
REGEL 150	232	REGEL 300	128
REGEL 160	119	REGEL 310	191
REGEL 170	95	REGEL 320	130
REGEL 180	240	REGEL 330	145
REGEL 190	12	REGEL 340	187
REGEL 200	6	REGEL 350	219
REGEL 210	184	REGEL 360	43
REGEL 220	43	REGEL 370	237
REGEL 230	89	REGEL 380	191
REGEL 240	154	REGEL 390	64

REGEL 400	56	REGEL 450	168
REGEL 410	55	REGEL 460	216
REGEL 420	140	REGEL 470	104
REGEL 430	27	REGEL 480	27
REGEL 440	36		

REFLEKTOR

Een van onze vaste inzenders van programma's voor deze rubriek is ongetwijfeld Fons Reijsbergen uit Leidschendam.
In een vierkant zitten 4 punten verstopt. Aan U de taak deze te vinden. Er moet een soort laserstraal in het vierkant gestuurd worden. Als U nu goed oplet waar deze het vierkant weer verlaat moet U deze punten kunnen vinden. De punten buigen de stralen namelijk af of absorberen ze. Een uitgebreide handleiding staat in het spel.
Een waarschuwing voor dit spel is wel op z'n plaats: geef ons niet de schuld als uw nachtrust hiervan te lijden heeft.

```
  1 rem reflektor / cbm-64
  2 rem door fons reijsbergen
  3 rem leidschendam / 070-278619
  4 rem
 10 rem ********** spel uitleg
 20 poke53280,0:poke53281,0:poke5327
    2,23
 30 dima$(23):gosub310
 40 rem ********** set up spel
 50 gosub740:gosub880
 60 rem ********** hoofdloop
 70 gosub960:sc=sc-1:ifft=3then50
 80 ifrtthen70
 90 ui=0:b=1:e=8:r=1:ifx=2orx=3thenb
    =8:e=1:r=-1
100 onxgosub190,250,190,250
110 a$=mid$(str$(y),2):ifui=5thena$=
    "--"
120 ifui=1thena$="A"+a$
130 ifui=2thena$="B"+a$
140 ifui=3thena$="C"+a$
150 ifui=4thena$="D"+a$
160 ifui=0thenprint"{CLR-HOME}ERROR{2xsh
    ift-SPATIE}ERROR":stop
170 a$(dp-1)=a$(dp-1)+a$:gosub1280:goto
    70
180 rem ********* laser vertikaal
190 fori=btoestepr:ifa(i,y)thenui=5:
    i=e:goto220
200 ifa(i+r,y+1)thenb=y:e=1:y=i:r=-1
    :goto250
210 ifa(i+r,y-1)thenb=y:e=8:y=i:r=1:
    goto250
220 next:ifui<>5thenui=1:ifr=1thenui
    =3
230 return
240 rem ********* laser horizontaal
250 fori=btoestepr:ifa(y,i)thenui=5:
    i=e:goto280
260 ifa(y+1,i+r)thenb=y:e=1:y=i:r=-1
    :goto190
270 ifa(y-1,i+r)thenb=y:e=8:y=i:r=1:
    goto190
280 next:ifui<>5thenui=4:ifr=1thenui
    =2
```

```
290 return
300 rem ********* uitleg
310 print"{CLR-HOME}{GEEL}{11xspatie}
    REFLEKTOR STRAAL"
320 print"{2xneer}{WIT}Bij dit spel
    is het de bedoeling om een viert
    al punten te vinden."
330 print"Deze punten bevinden zich
    in een vier-{2xspatie}kant van 8
    *8 vakjes."
340 print"{neer}Om deze{2xspatie}pun
    ten te vinden moet je vanaf"
350 print"de buiten kant een straal
    door het vier-kant sturen."
360 print"De straal gaat in een rech
    te lijn door{2xspatie}het vierka
    nt en wordt";
370 print" alleen door de{4xspatie}p
    unten weerkaatst."
380 print"{neer}Door nu goed te lett
    en op de in- en uit-gang van een
     straal, ";
390 print"moet je in staat{3xspatie}
    zijn de punten te vinden."
400 print"{neer}De straal kun je van
    af vanaf alle kantenhet vierkant
     insturen."
410 print"{neer}{GROEN}**Druk op een
     toets voor het vervolg.**{WIT}"
    :wait197,191
420 print"{CLR-HOME}Elke punt (*) ov
    erlapt de 8 omringende{2xspatie}
    vakjes, elk vakje heeft";
430 print" een ander effectop de str
    aal."
440 print"{neer}{GEEL}{4xspatie}A B
    C"
450 print" 0@2]2]2]2@"
460 print" ] ]]]]]]]{3xspatie}{WIT}s
    traal A word afgebogen"
470 print"{GEEL} +@[][][][@"
480 print" @@@=].]-@@{2xspatie}{WIT}
    straal B word geabsorbeerd"
490 print"{GEEL} +@[@[@[@[@"
500 print" ] ].]*].]{3xspatie}{WIT}s
    traal C word afgebogen"
510 print"{GEEL} +@[@[@[@[@"
515 print" ] ].].].]"
520 printtab(20)"[@[@[@[@["
530 print"{WIT}{2xspatie}Een kombina
    tie is {GEEL}]*] ] ] ]"
540 print"{WIT}{2xspatie}ook mogelij
    k.{5xspatie}{GEEL}[@[@[@[@["
550 printtab(20)"] ]0@@@@@ "
560 print"{WIT}{2xspatie}Dit is een
    simpel {GEEL}[@[][@[@["
570 print"{WIT}{2xspatie}voorbeeld.{7xsp
    atie}{GEEL}@@@=] ] ]"
580 printtab(20)"[@[@[@[@["
590 printtab(20)"] ] ]*] ]"
600 print"{2xneer}{GROEN}**Druk op e
    en toets voor het vervolg.**{WIT}"
    :wait197,191
610 print"{CLR-HOME}{neer}Je zend ee
    n straal het vierkant in door de
     letter van de";
620 print" rand en dan het cijfer va
    n het vakje in te typen."
630 print"{neer}Als je denkt te wete
    n waar de punten zyndruk dan op
    toets '' ";
640 print"en je mag het een gokje wa
    gen."
650 print"{neer}Als je gok fout is d
    an krijg je dat te{2xspatie}hore
    n, maar NIET wat er";
660 print" fout was."
670 print"{neer}Je mag maar drie kee
    r een poging wagen{2xspatie}om d
    e juiste plaatsen te";
680 print" raden."
690 print"Daarna ben je te laat en h
    eb je een{5xspatie}score van 0 p
    unten."
700 print"{2xneer}Je begint met een
    score van 100 bij elkestraal die
     je uitzendt ";
710 print"gaat er een punt vanaf."
720 print"{2xneer}{GROEN}{11xspatie}
    Druk op een toets.{WIT}":wait197
    ,191:return
730 rem ********* speelveld
740 print"{CLR-HOME}{neer}{6xspatie}
    1 2 3 4{neer}A{op}5 6 7 8{neer}"
```

```
750 printtab(5)"0@2@2@2@2@2@2@2@."
760 fori=1to7:print"{2xspatie}"i;tab(
    5)"] ] ] ] ] ] ] ]"i
770 ifi=4thenprinttab(4)"D+@[@[@[@[@
    [@[@[@3B":goto790
780 printtab(5)"+@[@[@[@[@[@[@[@3"
790 next:print"{3xspatie}8 ] ] ] ] ]
     ] ] ] ] 8":printtab(5)"-@1@1@1@
    1@1@1@1@="
800 print"{neer}{6xspatie}1 2 3 4{op}
    C{neer}5 6 7 8 "
810 a$="@@@@@@@@@@@@@@@@@@@@@@"
820 print"{HOME}{GROEN}{2xspatie}0"a
    $".0@@@@@@@@@@@@."
830 fori=1to21:print"{2xspatie}]"tab(
    24)"]]"tab(37)"]":next:print"{2xspat
    ie}+"a$"3]"tab(37)"]"
840 print"{2xspatie}]"tab(24)"]]"tab(
    37)"]"
850 print"{2xspatie}-"a$"=-@@@@@@@@@
    @@={WIT}";
860 sc=100:gosub1360:p=0:gosub1310:p
    =1:return
870 rem ********* nieuw spel
880 fori=1to8:forj=1to8:a(i,j)=0:next
    j,i:fori=1to23:a$(i)="":next
890 fori=1to4
900 x=int(rnd(1)*7+1):y=int(rnd(1)*7
    +1):ifa(x,y)=1then900
910 a(x,y)=1:next
920 b$="{HOME}":fori=0to22:b$=b$+"{neer}"
    :next:fori=0to20:b$=b$+"{rechts}"
    :next
930 dp=1:ft=0:rt=0
940 return
950 rem ********* input
960 printleft$(b$,27)"Straal in (bv
    B4) ":printb$;:poke204,0:rt=0
970 geta$:if(a$<"a"ora$>"d")anda$<>"
    "then970
980 ifa$=""thenpoke204,1:goto1090
990 i$=chr$(128+asc(a$)):printi$;:x=
    asc(a$)-64:
1000 geta$:if(a$<"1"ora$>"8")anda$<>chr$
     (20)then1000
1010 ifa$=chr$(20)thenprint" {2xlinks}
      {links}";:goto970
1020 printa$;:y=val(a$):j$=a$+" - "
1030 geta$:ifa$<>chr$(13)anda$<>chr$(
     20)then1030
1040 ifa$=chr$(20)thenprint" {2xlinks}
      {links}";:goto1000
1050 poke204,1:printb$"{3xspatie}"
1060 ifdp<22thena$(dp)=i$+j$:dp=dp+1:
     return
1070 fori=2to21:a$(i-1)=a$(i):next:a$
     (21)=i$+j$:return
1080 rem ******** raden van punten
1090 rt=1:g=0:fori=1to4:printleft$(b$
     ,27);"Punt"i"A. - D.{5xspatie}{11xl
     nks}";
1100 poke204,0
1110 geta$:ifa$<"1"ora$>"8"then1110
1120 printa$;"{4xrechts}";:x=val(a$)
1130 geta$:if(a$<"1"ora$>"8")anda$<>chr$
     (20)then1130
1140 ifa$=chr$(20)thenprint" {6xlinks}
      {links}";:goto1110
1150 printa$;:y=val(a$)
1160 geta$:ifa$<>chr$(13)anda$<>chr$(
     20)then1160
1170 ifa$=chr$(20)thenprint" {2xlinks}
      {links}";:goto1130
1180 ifa(y,x)=1theng=g+1:a(y,x)=2
1190 poke204,1:print" ":next
1200 ifg=4thengosub1310:printleft$(b$
     ,27)"GOED ZO! SCORE"sc"{links} "
     :ft=3:goto1250
1210 fori=1to8:forj=1to8:ifa(i,j)=2then
     a(i,j)=1
1220 nextj,i
1230 ft=ft+1:ifft<3thenprintleft$(b$,
     27);"JAMMER"4-g"{links} NIET GOE
     D.":goto1260
1240 printleft$(b$,27);"NIET GERADEN
     IN 3 x"
1250 gosub1310
1260 gosub1360:wait197,191:return
1270 rem ******** print in-uit tabel
1280 gosub1360
1290 print"{HOME}{2xneer}":fori=1to21
     :printtab(27)a$(i):next:print"{2xop}"
     :return
1300 rem ******** print de punten
1310 print"{HOME}{3xneer}{GEEL}":fori
     =1to8:printtab(6);:forj=1to8
1320 ifa(i,j)<>0andp=1thenprint"*{rechts}"
     ;:goto1340
1330 print"{RVS-aan} {RVS-uit}{rechts}"
     ;
1340 next:print"{neer}":next:print"{WIT}"
     :return
1350 rem ******** print score
1360 ifsc<0thensc=0
```


```
1310 print"{HOME}{3xneer}{GEEL}":fori
     =1to8:printtab(6);:forj=1to8
1320 ifa(i,j)<>0andp=1thenprint"*{rechts}"
     ;:goto1340
1330 print"{RVS-aan} {RVS-uit}{rechts}"
     ;
1340 next:print"{neer}":next:print"{WIT}"
     :return
1350 rem ******** print score
1360 ifsc<0thensc=0
```


```
1370 print"{HOME}{neer}"tab(27)"{l.groen}
     SCORE"sc"{links} {WIT}":return
```

*** EINDE LISTING ***

REGEL 1	213	REGEL 80	63
REGEL 2	41	REGEL 90	244
REGEL 3	36	REGEL 100	92
REGEL 4	143	REGEL 110	164
REGEL 10	49	REGEL 120	222
REGEL 20	138	REGEL 130	224
REGEL 30	252	REGEL 140	226
REGEL 40	248	REGEL 150	228
REGEL 50	143	REGEL 160	64
REGEL 60	221	REGEL 170	201
REGEL 70	112	REGEL 180	35

REGEL 190	196
REGEL 200	26
REGEL 210	119
REGEL 220	202
REGEL 230	142
REGEL 240	203
REGEL 250	202
REGEL 260	29
REGEL 270	122
REGEL 280	204
REGEL 290	142
REGEL 300	211
REGEL 310	3
REGEL 320	237
REGEL 330	73
REGEL 340	153
REGEL 350	175
REGEL 360	248
REGEL 370	78
REGEL 380	111
REGEL 390	222
REGEL 400	221
REGEL 410	35
REGEL 420	205
REGEL 430	233
REGEL 440	210
REGEL 450	108
REGEL 460	12
REGEL 470	169
REGEL 480	30
REGEL 490	82
REGEL 500	253
REGEL 510	82
REGEL 515	184
REGEL 520	82
REGEL 530	210
REGEL 540	48
REGEL 550	148
REGEL 560	131
REGEL 570	67
REGEL 580	82
REGEL 590	134
REGEL 600	52
REGEL 610	114
REGEL 620	33
REGEL 630	29
REGEL 640	157
REGEL 650	154
REGEL 660	52
REGEL 670	113
REGEL 680	117
REGEL 690	133
REGEL 700	71
REGEL 710	74
REGEL 720	8
REGEL 730	173
REGEL 740	153
REGEL 750	26
REGEL 760	118
REGEL 770	130
REGEL 780	57
REGEL 790	39
REGEL 800	247
REGEL 810	27
REGEL 820	179
REGEL 830	61
REGEL 840	65
REGEL 850	218
REGEL 860	151
REGEL 870	197
REGEL 880	16
REGEL 890	133
REGEL 900	168
REGEL 910	14
REGEL 920	59
REGEL 930	239
REGEL 940	142
REGEL 950	153
REGEL 960	6
REGEL 970	105
REGEL 980	3
REGEL 990	130
REGEL 1000	69
REGEL 1010	21
REGEL 1020	211
REGEL 1030	109
REGEL 1040	54
REGEL 1050	7
REGEL 1060	21
REGEL 1070	137
REGEL 1080	8
REGEL 1090	109
REGEL 1100	137
REGEL 1110	4
REGEL 1120	235
REGEL 1130	73
REGEL 1140	172
REGEL 1150	249
REGEL 1160	113
REGEL 1170	58
REGEL 1180	44
REGEL 1190	93
REGEL 1200	124
REGEL 1210	34
REGEL 1220	65
REGEL 1230	165
REGEL 1240	244
REGEL 1250	82
REGEL 1260	83
REGEL 1270	138
REGEL 1280	87
REGEL 1290	196
REGEL 1300	207
REGEL 1310	88
REGEL 1320	227
REGEL 1330	217
REGEL 1340	74
REGEL 1350	232
REGEL 1360	35
REGEL 1370	83

OP Z'N KOP

We weten allemaal dat er met onze trouwe 64 er allerlei (vreemde) beeldeffecten mogelijk zijn.
Letters op zijn kop zetten is niet zo moeilijk maar dat dan alles ook nog werkt! Ook de cursor besturing is omgedraaid.
Dit is een ideaal programma om even te runnen in een computershop om de mensen eens goed aan het schrikken te brengen.

```
100 sys65409:print"{2xneer}"tab(15)"
    op z'n kop"
110 print"{2xneer} na 'sys 49152' wo
    rdt heel het scherm"
120 print"{neer} omgedraaid. je kunt
     dan nog gewoon"
130 print"{neer} typen en zelfs prog
    rammeren!!!"
140 a=49152:c=0
150 readx$:fori=1tolen(x$)/2:x=10*(asc
    (mid$(x$,i*2-1))-65)+asc(mid$(x$
    ,i*2))-48
160 pokea,x:c=c+x:a=a+1:next:ifa<493
    90goto150
170 print"{neer}{2xrechts}druk op ee
    n toets..."
180 poke198,0:wait198,1:poke198,0:sys
    65409:sys49152:sys58260
190 data m0q9f1n3a1q9a0n3h5n3h7q9
200 data u8n3h6q9w4n3h8q9a0n3h9q9
210 data a7n3i0q4h9r7h5n3i4q9a0n3
220 data i3q9a1n3i1q9m8n3i2q5i4d7
230 data i1y0a6q5i3a5i2n3i3a6i1h0
240 data i2o4x8q5i3q4i0o5h7x0h9t8
250 data i0b6u8q5h5c4k5a8n3h5q5h6
260 data k5a0n3h6q5h7c4k5a8n3h7q5
270 data h8k5a0n3h8u1y0u8r0q9f5n3
280 data a1q9d2q2a0p7a0y0p7a0y1p7
290 data a0y2p7a0y3x2u8y1r3a2w1a9
300 data a3o1a2w1r3a0w1e1z2o1a0w1
310 data q9u1o1c4u8q2q5q0t2o2c0a3
320 data o0c1a3q9a0n3a2i8j6t8a2b6
330 data g6q9a3n3a2q2a0q0a4n4z1n2
340 data z2q2x1q0y3n4z3n2z4q0a0r7
350 data z1o5z3x0z1u8a2x0z2q5z3u8
360 data a2t8z4t8z3q5z4u1y0r6x2r3
370 data n4a2q2a0p7a0v6p7a0v7p7a0
380 data v8p7a0v9x2u8y1h6e9x4

*** EINDE LISTING ***
```

ôREGEL 100	173
REGEL 110	49
REGEL 120	64
REGEL 130	135
REGEL 140	87
REGEL 150	84
REGEL 160	182
REGEL 170	238
REGEL 180	201
REGEL 190	119
REGEL 200	172
REGEL 210	123
REGEL 220	127
REGEL 230	86
REGEL 240	169
REGEL 250	118
REGEL 260	110
REGEL 270	153
REGEL 280	115
REGEL 290	158
REGEL 300	107
REGEL 310	143
REGEL 320	81
REGEL 330	105
REGEL 340	192
REGEL 350	230
REGEL 360	227
REGEL 370	109
REGEL 380	193

TOPOGRAFIE SPEL

Marc de Hingh heeft voor ons een spel gemaakt kwaarin de topografie van Nederland centraal staat. Het bijzondere is echter dat het plaatsnamen-bestand zelf eenvoudig is uit te breiden, door de plaatsnaam in te typen en met de cursor de plaats aan te wijzen. Deze wordt dan in dataregels weggeschreven.
Niet vergeten de nieuwe versie van het programma ook weg te schrijven.

```
100 poke55,128:poke56,62:clr
110 c=0:fora=16000to16463:readd:poke
    a,d:c=c+d:next
120 ifc<>48154thenprint"fout in data
     regel 9000 en verder":end
130 fori=0to3:poke16464+i,peek(63+i)
    :next
140 x=78:y=80:i=rnd(-ti):vp$="utrech
    t"
150 sys65409:v=53248:pokev+21,255:poke
    v+23,255:pokev+27,255:pokev+28,0

160 pokev+29,255:fori=0to5:poke2040+
    i,250+i:pokev+39+i,1:next:sys163
    84,x,y
170 print"{CLR-HOME}{2xneer}"tab(18)
    "topografie-spel"
180 printtab(18)"{2xneer}1-{2xspatie}
    start spel":printtab(18)"{neer}2
    -{2xspatie}plaatsenbestand"
190 printtab(18)"{neer}3-{2xspatie}o
    pzoeken":printtab(18)"{neer}4-{2xspa
    tie}stop"
200 printtab(18)"{2xneer}maak uw keu
    ze"
210 gett$:onval(t$)goto220,710,570,9
    999:goto210
220 gosub1170:print"{HOME}{4xneer}"tab(
    20)"klaar voor de start"
230 poke1472,43:poke55744,0:sp=15:sc
    =0
240 fori=0to190step3:tt=i:gosub1150:
    next
250 printtab(20)"{op}{19xspatie}"
260 gosub1190
270 readaa:fori=0tornd(1)*aa:readp$,
    xx,yy:next:i=fre(0)
280 ifp$=vp$then260
290 print"{HOME}{5xneer}"tab(19)"ga
    naar "p$:printtab(18)"{neer}gebr
    uik de cursor"
300 printtab(18)"toetsen of joystick
     2":printtab(20)"{neer}druk dan
    op return"
310 printtab(22)"of vuurknop":poke19
    8,0
320 tt=tt-1:gosub1150:iftt<1then520
330 fori=1tosp
340 gett$:jy=peek(56320)and31:ift$=""
    andjy=0goto320
350 ift$=chr$(13)or((jyand16)=0)then
    410
360 if(t$="{op}"or((jyand1)=0))andy<
    150theny=y+1
370 if(t$="{neer}"or((jyand2)=0))and
    y>20theny=y-1
380 if(t$="{links}"or((jyand4)=0))and
    x<125thenx=x+1
390 if(t$="{rechts}"or((jyand8)=0))and
    x>32thenx=x-1
400 sys16384,x,y:next:goto320
410 ifabs(x-xx)<3andabs(y-yy)<3thensys
    16384,xx,yy:x=xx:y=yy:goto490
420 printtab(23)"{2xneer}u zit er zo
    'n"
430 printtab(20)int(rnd(1)+2.5*sqr((
    x-xx)^2+(y-yy)^2))"km naast..."
440 fori=1to1000:next
450 x=x+sgn(xx-x):y=y+sgn(yy-y):sys1
    6384,x,y
460 tt=tt-1:gosub1150:iftt<1then520
470 ifx<>xxory<>yythen450
480 goto510
490 printtab(25)"{2xneer}goed!":sc=s
    c+10:printtab(20)"{2xneer}score:"
    sc
500 fori=1to15-sp/2:tt=tt+1:gosub115
    0:iftt<196thennext
510 sp=sp-1:fori=1to2000:next:gosub1
    170:vp$=p$:goto260
520 gosub1170:print"{HOME}{3xneer}"tab(
    22)"game over"
530 printtab(20)"{2xneer}uw score:"s
    c
540 printtab(19)"{4xneer}druk op 'n
    toets":poke198,0
550 gett$:ift$=""goto550
560 run140
570 gosub1170:print"{HOME}{3xneer}"tab(
    19)"welke plaats wilt u":printtab(
    20)"opzoeken?{neer}"
580 printtab(20):inputpl$
590 gosub1190
600 readaa:fori=1toaa:readp$,xx,yy:if
    p$<>pl$thennext:goto660
610 x=x+sgn(xx-x):y=y+sgn(yy-y):sys1
    6384,x,y
620 ifx<>xxory<>yythen610
630 printtab(19)"{3xneer}de '+' wijs
    t waar":printtab(20)pl$" ligt."
640 fori=1to10:poke55744,0:poke1472,
    32:forj=1to50:next:poke1472,43
650 forj=1to100:next:next:goto670
660 printtab(19)"{3xneer}die plaats
    komt niet":printtab(19)"voor in
    het bestand."
670 printtab(19)"{2xneer}wilt u nog
    'n plaats":printtab(22)"opzoeken
    ? (j/n)"
680 gett$:ift$="n"thenrun140
690 ift$<>"j"then680
700 goto570
710 print"{CLR-HOME}{2xneer}"tab(18)
    "plaatsenbestand":poke53265,27
720 printtab(18)"{2xneer}1-{2xspatie}
    lijst naar scherm"
730 printtab(18)"{neer}2-{3xspatie}'
    '{3xspatie}''{2xspatie}printer"
740 printtab(18)"{neer}3-{2xspatie}n
```

```
     ieuwe toevoegen"
 750 printtab(18)"{neer}4-{2xspatie}o
     ude wijzigen"
 760 printtab(18)"{neer}5-{2xspatie}t
     erug naar menu"
 770 printtab(18)"{2xneer}maak uw keu
     ze":poke198,0
 780 gett$:onval(t$)goto790,910,930,1
     090,170:goto780
 790 gosub1190:i=1:readaa:print"{CLR-HOME
     }"
 800 readp$,x,y:printtab(18)i;p$
 810 poke1472,43:poke55744,0:sys16384
     ,x,y
 820 ifi=aathen880
 830 printtab(19)"{neer}return voor v
     olgende":poke198,0
 840 gett$:ift$=""goto840
 850 ift$<>chr$(13)then710
 860 i=i+1:ifpeek(214)>23thenprint"{CLR-H
     OME}"
 870 printtab(19)"{op}{20xspatie}{2xop}"
     :goto800
 880 printtab(19)"{neer}einde, druk '
     n toets":poke198,0
 890 gett$:ift$=""goto890
 900 goto710
 910 gosub1190:readaa:open4,4:fori=1to
     aa:readp$,xx,yy
 920 print#4,i;p$:next:close4:goto710

 930 gosub1190:readaa:aa=aa+1:i=aa
 940 print"{CLR-HOME}{3xneer}"tab(19)
     "plaats toevoegen aan":printtab(
     19)"bestand{neer}"
 950 printtab(20)"plaatsnaam?":printtab(
     20):inputpl$
 960 printtab(20)"{2xneer}geef nu de
     ligging":printtab(20)"van de pla
     ats aan"
 970 printtab(21)"gebruik crsr en":print
     tab(21)"return."
 980 poke1472,43:poke55744,0
 990 gett$:ift$=""then990
1000 ift$=chr$(13)then1060
1010 ift$="{op}"andy<150theny=y+1
1020 ift$="{neer}"andy>20theny=y-1
1030 ift$="{links}"andx<125thenx=x+1
1040 ift$="{rechts}"andx>32thenx=x-1
1050 sys16384,x,y:goto990
1060 poke53265,11:print"{CLR-HOME}{2xneer
     }10000 data"aa:print10000+i"data
      "pl$","x","y
1070 print"x="x":y="y":goto710{HOME}"
     ;
1080 poke198,3:poke631,13:poke632,13:
     poke633,13:end
1090 print"{CLR-HOME}{3xneer}"tab(19)
     "plaatsnaam of coord.":printtab(
     19)"veranderen."
1100 printtab(20)"{neer}oude plaatsna
     am?":printtab(20):inputpl$
1110 gosub1190:readaa:fori=1toaa:read
     p$,xx,yy
1120 ifp$<>pl$thennext:printtab(20)"{2xne
     er}onbekend.":fori=1to1000:next:
     goto710
1130 x=xx:y=yy:sys16384,x,y:poke1472,
     43:poke55744,0
1140 printtab(20)"nieuwe ";:goto950
1150 s=int(tt/8)*40:bi=7andtt:poke562
     57-s,8:ifsthenpoke2025-s,160
1160 poke1985-s,peek(16456+bi):return

1170 print"{HOME}":fori=1to23:printtab(
     18):poke19,4:printspc(21):poke19
     ,0:print
1180 next:return
1190 fori=0to3:poke63+i,peek(16464+i)
     :next:return
9000 data0,0,2,0,0,48,0,46,1,3,192,18
     5,8,3,207,16,15,255,0,31,255,32,
     63,255,96
9010 data63,255,96,127,255,64,127,255
     ,0,63,255,64,127,255,124,127,255
     ,252,3,255
9020 data252,15,255,255,3,255,255,15,
     255,254,30,127,248,1,255,248,29,
     255,0,0,0
9030 data0,0,0,0,224,0,0,224,0,0,240,
     0,0,248,0,0,252,0,0,254,0,0,254,
     0,0,254,0
9040 data0,254,0,0,254,0,0,254,0,0,25
     2,0,0,252,0,0,252,0,0,252,0,0,25
     2,0,0,192
9050 data0,0,192,0,0,128,0,0,0,0,15,1
     95,0,31,199,0,31,223,0,31,159,0,
     31,206,0
9060 data63,241,0,63,255,0,127,255,0,
     127,255,0,255,255,1,255,255,3,25
     5,255,7
9070 data255,255,15,255,255,7,255,255
     ,3,255,255,56,255,255,6,127,255,
     243,7,255
9080 data156,127,255,11,255,255,0,223
     ,248,0,223,254,0,191,255,0,127,2
     55,192,255
9090 data255,192,255,255,192,255,255,
     128,255,254,0,255,252,0,255,224,
     0,255,224
9100 data0,255,232,0,255,248,0,255,24
     8,0,251,192,0,252,0,0,248,0,0,25
     2,0,0,252
9110 data0,0,254,0,0,254,0,0,0,54,63,
     255,120,111,255,63,15,223,23,253
     ,219,98,12
9120 data3,249,208,3,255,224,1,103,19
     2,0,3,0,0,0,0,0,0,0,0,0,0,0,0,
     0,0,0,0,0
9130 data0,0,0,0,0,0,0,0,0,0,0,0,0,0,
     0,0,0,0,0,0,0,255,224,0,255,240,
     0,255,240
9140 data0,255,240,0,255,240,0,255,24
     0,0,255,224,0,15,224,0,15,192,0,
     1,240,0,1
9150 data224,0,1,192,0,1,128,0,3,192,
     0,3,224,0,3,224,0,7,192,0,3,192,
     0,0,0,0,0
9160 data0,0,0,0,0,0,32,241,183,120,1
     73,17,208,16,251,138,141,0,208,2
```

```
      4,105,48
 9170 data141,2,208,56,233,72,141,4,20
      8,24,105,48,141,6,208,56,233,56,
      141,8,208
 9180 data24,105,48,141,10,208,32,241,
      183,138,141,1,208,141,3,208,24,1
      05,42,141
 9190 data5,208,141,7,208,105,42,141,9
      ,208,141,11,208,88,96,32,100,111
      ,121,98
 9200 data248,247,227
 9999 sys65409:end
10000 data 21
10001 data amsterdam, 82 , 92
10002 data den haag, 99 , 79
10003 data utrecht, 78 , 80
10004 data breda, 87 , 57
10005 data rotterdam, 94 , 71
10006 data groningen, 45 , 128
10007 data leeuwarden, 62 , 126
10008 data assen, 43 , 118
10009 data maastricht, 64 , 24
10010 data heerlen, 57 , 27
10011 data venlo, 54 , 47
10012 data eindhoven, 69 , 51
10013 data dordrecht, 91 , 65
10014 data vlissingen, 117 , 51
10015 data nijmegen, 61 , 69
10016 data arnhem, 57 , 75
10017 data apeldoorn, 57 , 86
10018 data haarlem, 90 , 93
10019 data almere, 77 , 93
10020 data hilversum, 77 , 86
10021 data leiden, 93 , 83
19999 rem
20000 rem bovenstaande plaatsnamen zij
      n voorbeelden. het plaatsenbesta
      nd
20001 rem kan gemakkelijk vanuit het p
      rogramma aangevuld/veranderd wor
      den.

 *** EINDE LISTING ***
```

```
öREGEL 100    110      REGEL 300    73
REGEL 110     21       REGEL 310    251
REGEL 120     248      REGEL 320    251
REGEL 130     163      REGEL 330    244
REGEL 140     200      REGEL 340    83
REGEL 150     129      REGEL 350    90
REGEL 160     249      REGEL 360    201
REGEL 170     92       REGEL 370    21
REGEL 180     105      REGEL 380    215
REGEL 190     242      REGEL 390    39
REGEL 200     126      REGEL 400    193
REGEL 210     129      REGEL 410    205
REGEL 220     254      REGEL 420    49
REGEL 230     213      REGEL 430    82
REGEL 240     237      REGEL 440    206
REGEL 250     156      REGEL 450    174
REGEL 260     88       REGEL 460    251
REGEL 270     23       REGEL 470    86
REGEL 280     186      REGEL 480    31
REGEL 290     170      REGEL 490    72
```

```
REGEL 500    114       REGEL 1150    201
REGEL 510    242       REGEL 1160    174
REGEL 520    125       REGEL 1170    242
REGEL 530    37        REGEL 1180    74
REGEL 540    255       REGEL 1190    107
REGEL 550    111       REGEL 9000    175
REGEL 560    31        REGEL 9010    27
REGEL 570    255       REGEL 9020    207
REGEL 580    70        REGEL 9030    89
REGEL 590    88        REGEL 9040    105
REGEL 600    234       REGEL 9050    80
REGEL 610    174       REGEL 9060    103
REGEL 620    84        REGEL 9070    242
REGEL 630    103       REGEL 9080    255
REGEL 640    247       REGEL 9090    222
REGEL 650    187       REGEL 9100    159
REGEL 660    130       REGEL 9110    214
REGEL 670    241       REGEL 9120    74
REGEL 680    96        REGEL 9130    59
REGEL 690    58        REGEL 9140    141
REGEL 700    37        REGEL 9150    115
REGEL 710    207       REGEL 9160    106
REGEL 720    252       REGEL 9170    201
REGEL 730    66        REGEL 9180    176
REGEL 740    252       REGEL 9190    97
REGEL 750    24        REGEL 9200    177
REGEL 760    99        REGEL 9999    96
REGEL 770    77        REGEL 10000    230
REGEL 780    69        REGEL 10001    78
REGEL 790    171       REGEL 10002    165
REGEL 800    4         REGEL 10003    209
REGEL 810    21        REGEL 10004    20
REGEL 820    79        REGEL 10005    98
REGEL 830    109       REGEL 10006    134
REGEL 840    113       REGEL 10007    200
REGEL 850    34        REGEL 10008    86
REGEL 860    187       REGEL 10009    155
REGEL 870    33        REGEL 10010    179
REGEL 880    190       REGEL 10011    51
REGEL 890    118       REGEL 10012    80
REGEL 900    33        REGEL 10013    79
REGEL 910    4         REGEL 10014    220
REGEL 920    21        REGEL 10015    254
REGEL 930    29        REGEL 10016    110
REGEL 940    189       REGEL 10017    89
REGEL 950    172       REGEL 10018    170
REGEL 960    185       REGEL 10019    107
REGEL 970    54        REGEL 10020    118
REGEL 980    46        REGEL 10021    99
REGEL 990    149       REGEL 19999    143
REGEL 1000    159      REGEL 20000    21
REGEL 1010    193      REGEL 20001    68
REGEL 1020    12
REGEL 1030    204
REGEL 1040    24
REGEL 1050    18
REGEL 1060    233
REGEL 1070    212
REGEL 1080    70
REGEL 1090    158
REGEL 1100    234
REGEL 1110    151
REGEL 1120    138
REGEL 1130    0
REGEL 1140    116
```

BLACKJACK

Een overbekend spel waarvan eigenlijk iedereen de spelregels wel kent. Zo niet dan worden ze U nog eens uitgelegd na dat u het programma heeft ingetypt.

```
  1 rem    ***blackjack***
  2 rem    door michael burghout
  3 rem    uit haarlem  / 023 330454
  4 rem    da costastr. 26
  5 rem *****************************
    *
  6
  7 print"{3xspatie}{RVS-aan}{4xspatie}
    instructies{5xspatie}{RVS-uit}"
  8 print:print" de bedoeling v.h sp
    el is 21 te krijgen"
  9 print" een boer ,vrouw en koning
     zijn 10 "
 10 print" een aas is 11 of 1{2xspatie}"

 11 print" de getallen zijn hun eige
    n waarde"
 12 print:print"{7xspatie}veel plez
    ier{6xspatie}"
 13 for a=1to 5000:next a
 14
 15 bl$="?????{neer}{6xlinks}":cl$=
    "{6xspatie}{neer}{6xlinks}":cn$=
    "{HOME}{13xneer}"
 20 ca$="{17xrechts}":nl$="{22xspatie}"

 30 c$="CCCCCC":d$="B{neer}{links}B{neer
    }{links}B{neer}{links}B{neer}{links}
    B{neer}{links}B{neer}{links}B{neer}
    {links}B":cr=500:dc$="{2xneer}"
 40 8:dimss$(3):dimpc(9):dimcc(9)
 50 dimcd$(12):dimcn%(12):fori=0to12
    :readcd$(i):cd$(i)=cd$(i)+" ":cn
    %(i)=15:nexti
 60 dataa,2,3,4,5,6,7,8,9,10,j,q,k
 70 cd$(9)="10":ss$(0)="{ROOD}S":ss$
    (1)="{ROOD}Z":ss$(2)="{ZWART}X":
    ss$(3)="{ZWART}A"
 80 print"{CLR-HOME}{ROOD}{15xspatie}
    blackjack"
 90 print"{l.rood}{2xneer}{rechts}c{neer
    }{links}o{neer}{links}m{neer}{links}
    p{neer}{links}u{neer}{links}t{neer}
    {links}e{neer}{links}r{7xneer}{links
    }{BLAUW}u"
100 l=0:cx$="{4xrechts}":sq=0:b=0:cn
    =-1:pn=-1:db=0:s=0
110 cy$=cn$+dc$:gosub320:p1$=dt$:ps=
    x:pn=0:pc(pn)=y:gosub420
120 cy$=dc$:gosub320:c1$=dt$:cs1=x:c
    n=0:cc(cn)=y
130 gosub560:cy$=cn$+dc$:dt$=p1$:x=p
    s:gosub380:rc$=cx$:rp$=cx$
140 printcn$;ca$;"{l.blauw} inzet{ZWART}"
    ;:inputb
150 printcn$;ca$;nl$
160 ifb<1orb>crthen140
170 gosub550
180 cy$=cn$+dc$:gosub320:pn=pn+1:pc(
    pn)=y:p1$=dt$:ps=x
190 cy$=dc$:cx$=rc$:gosub320:c2$=dt$
    :cs2=x:cn=cn+1:cc(cn)=y
200 cy$=cn$+dc$:cx$=rp$:dt$=p1$:x=ps
    :gosub380:gosub420
210 ifpv=21then290
220 printcn$;ca$;"{bruin}stoppen of
    kaart";
230 geta$:ifa$=""then230
240 ifa$<>"s"anda$<>"k"anda$<>"d"then
    printcn$;ca$;nl$:directory1,350,
    5:goto210
250 ifa$="k"thengosub320:pn=pn+1:pc(
    pn)=y:gosub380:gosub420:gosub600
    :goto210
260 ifa$<>"d"ordb>0orb>crthen280
270 gosub550:db=1:goto210
280 ifa$<>"s"orpv<12then210
290 s=1:gosub570:gosub480
300 gosub600
310 gosub320:cn=cn+1:cc(cn)=y:gosub3
    80:gosub480:goto300
320 directory3,700,5:y=int(rnd(0)*13
    ):ifcn%(y)=0then320
330 dt$=cd$(y):x=int(rnd(0)*4)
340 if(cn%(y)and(2^x))=0then320
350 cn%(y)=cn%(y)-2^x
360 print"{HOME}{ZWART}";cy$;cx$;"U"
    ;c$;"I{neer}{links}";d$;"{HOME}"
    ;cy$;cx$;"{neer}";d$;"{neer}{links}
    J";c$;"K";
370 print"{HOME}";cy$;cx$;"{neer}{rechts
    }";bl$;bl$;bl$;bl$;bl$;bl$;bl$;b
    l$;:fori=1to100:next:return
380 cc$=left$(ss$(x),1)
390 fori=1to500:next:print"{HOME}";c
    y$;cx$;"{neer}{rechts}";cc$;dt$;
    "{4xspatie}{neer}{6xlinks}";ss$(
    x);"{5xspatie}{neer}{6xlinks}";
400 printcl$;cl$;cl$;cl$;cl$;cl$;:cx
    $=cx$+"{3xrechts}"
410 directory1,1000,3:fori=1to500:next
    :return
420 pv=0:ac=0:fori=0topn:pv=pv+pc(i)
    +1:ifpc(i)=0thenpv=pv+10:ac=ac+1

430 ifpc(i)>9thenpv=pv-pc(i)+9
440 next
450 ifpv>21andac>0thenpv=pv-10:ac=ac
    -1:goto450
460 ifpv>21thenl=1
470 return
480 ifcn<0thenreturn
490 cv=0:ac=0:fori=0tocn:cv=cv+cc(i)
    +1:ifcc(i)=0thencv=cv+10:ac=ac+1

500 ifcc(i)>9thencv=cv-cc(i)+9
510 next
520 ifcv>21andac>0thencv=cv-10:ac=ac
    -1:goto520
530 ifcv>21thenl=-1
540 return
550 sq=sq+b:printcn$;"{oranje}{neer}
```

```
   {rechts}ingezet ¦{8xspatie}{8xlinks}
   {grijs 2}";sq:cr=cr-b
560 printcn$;"{oranje}{rechts}bedrag
    ¦{8xspatie}{8xlinks}{grijs 2}";
   cr:return
570 print"{HOME}{3xneer}{5xrechts}";
   c1$;"{neer}{2xlinks}";ss$(cs1);"
    {neer}{2xlinks}{2xspatie}{neer}
   {2xlinks}{2xspatie}{neer}{2xlinks}
   {2xspatie}{neer}{2xlinks}{2xspatie}
   {neer}{2xlinks}{2xspatie}{neer}{2xli
   nks}{2xspatie}"
580 directory1,1000,3
590 cy$=dc$:cx$=rc$:dt$=c2$:x=cs1:gosub
   380:gosub480:return
600 ifl=1thenprintcn$;ca$;"{ROOD} je
    bent dood !{3xspatie}":directory
   1,800,10:directory1,700,20:goto7
   80
610 ifl<>-1then640
620 printcn$;ca$;nl$;cn$;ca$;"{PURPER}
   {2xspatie}ik ben dood !":cr=cr+2
   *sq
630 directory1,800,10:fori=1to50:next
   :directory1,900,10:goto790
640 ifcv<>21orcn<>1then670
650 printcn$;ca$;"{ZWART}ik heb bla
   ckjack!":directory1,900,20:ifpv
   =21andpn=1thencr=cr-3*sq
660 fori=1to10:directory1,800+10*i,1
   0:next:cr=cr-sq:goto790
670 ifs=0thenreturn
680 ifpv<>21orpn<>1then710
690 printcn$;ca$;"{ZWART}jij heb bl
   ackjack!":cr=cr+3*sq:gosub570
700 fori=1to30:directory1,300+10*i,1
   0:directory2,420+10*i,10::forj=1
   to50:nextj:nexti:goto790
710 ifcv<17thenprintcn$;ca$;"{ROOD}i
   k neem een kaart ":return
720 ifcv<pvthen740
730 printcn$;ca$;"{grijs 2}{4xspatie}
   jij verliest":directory1,700,10:
   fori=1to50:next:directory1,600,2
   5:goto790
740 printcn$;ca$;"{GROEN}{4xspatie}i
   k verlies{4xspatie}"
750 fori=1to10:directory1,900+10*i,1
   0:forj=1to50:nextj:directory1,10
   00-10*i,10:next
760 cr=cr+2*sq
770 goto790
780 gosub570
790 gosub560:printcn$;"{neer}{rechts}"
   ;nl$;cn$;ca$;"{GROEN}{neer} nog
   een spel ? j/n{2xspatie}"
800 geta$:ifa$=""then800
810 ifa$="n"thenend
820 ifcr>0then80
830 printcn$;ca$;"{l.blauw}{4xspatie}
   platzak !":end

*** EINDE LISTING ***
```

REGEL 1	1	REGEL 360	220
REGEL 2	38	REGEL 370	213
REGEL 3	114	REGEL 380	69
REGEL 4	29	REGEL 390	175
REGEL 5	81	REGEL 400	220
REGEL 6	232	REGEL 410	251
REGEL 7	222	REGEL 420	141
REGEL 8	100	REGEL 430	2
REGEL 9	161	REGEL 440	130
REGEL 10	78	REGEL 450	54
REGEL 11	244	REGEL 460	27
REGEL 12	253	REGEL 470	142
REGEL 13	11	REGEL 480	52
REGEL 14	232	REGEL 490	37
REGEL 15	88	REGEL 500	206
REGEL 20	121	REGEL 510	130
REGEL 30	108	REGEL 520	13
REGEL 40	206	REGEL 530	185
REGEL 50	13	REGEL 540	142
REGEL 60	199	REGEL 550	112
REGEL 70	28	REGEL 560	132
REGEL 80	2	REGEL 570	175
REGEL 90	107	REGEL 580	87
REGEL 100	127	REGEL 590	90
REGEL 110	234	REGEL 600	192
REGEL 120	30	REGEL 610	88
REGEL 130	73	REGEL 620	10
REGEL 140	160	REGEL 630	2
REGEL 150	42	REGEL 640	5
REGEL 160	37	REGEL 650	57
REGEL 170	39	REGEL 660	204
REGEL 180	214	REGEL 670	245
REGEL 190	192	REGEL 680	26
REGEL 200	224	REGEL 690	47
REGEL 210	136	REGEL 700	115
REGEL 220	177	REGEL 710	126
REGEL 230	98	REGEL 720	191
REGEL 240	131	REGEL 730	209
REGEL 250	54	REGEL 740	124
REGEL 260	108	REGEL 750	188
REGEL 270	32	REGEL 760	8
REGEL 280	145	REGEL 770	41
REGEL 290	252	REGEL 780	41
REGEL 300	35	REGEL 790	205
REGEL 310	211	REGEL 800	101
REGEL 320	11	REGEL 810	91
REGEL 330	41	REGEL 820	16
REGEL 340	146	REGEL 830	66
REGEL 350	85		

TEKENING 1 en 2

Deze programma's zijn ingezonden door: .....helaas was er geen briefje bij. We plaatsen dit om te laten zien wat er met een heel klein programma voor de C-16 mogelijk is.

```
tekening 1
 1 0:0,1:1,2:4,1
10 1,1
40 0,100
50 forx=0to316step3
```

PRINT-OUT · PRINT-OUT · PRINT-OUT · PRINT-OUT · PRINT-OUT · PRINT-OUT · PRINT-OUT · PRINT-OUT

```
 60 y=int(100+100*sin(x/40))
 70 1tox,yto1,100to65,190 :nextx

*** EINDE LISTING ***


 ôREGEL 1    30
 REGEL 10    108
 REGEL 40    211
 REGEL 50    213
 REGEL 60    2
 REGEL 70    18
```

```
tekening 2

 1 0:0,1:1,2:4,1
10 1,1
40 0,100
50 forx=0to316step3
60 y=int(100+100*sin(x/40))
70 1tox,yto1,50:nextx

*** EINDE LISTING ***

 ôREGEL 1    30      REGEL 50    213
 REGEL 10    108     REGEL 60    2
 REGEL 40    211     REGEL 70    17
```

PRINT-OUT C-16 · PRINT-OUT C-16 · PRINT-OUT C-16 · PRINT-OUT C-16 · PRINT-OUT C-16

```
checksum c16

 10 rem *****************************
    *****
 20 rem syntax.checksum
 30 rem voor c-16 & plus/4
 40 rem
 50 rem syntax testen met 'sys 1536'

 60 rem
 70 rem v.851128.16       jan bodzing
    a
 80 rem *****************************
    *****
 90 i=1536       :rem beginadres
100 reada:ifa>=0then pokei,a:i=i+1:goto
    100
110 print"data is weggezet"
120 print"cheksum printen met 'sys 1
    536'
130 end
200 data 165, 43,166, 44,133
210 data  31,134, 32,169,147
220 data  32,210,255,160,  0
230 data 240,  3, 32, 73,  6
240 data  32, 73,  6,208,  1
250 data  96, 72,152, 32,131
260 data   6,168,104,234, 32
270 data  81,  6, 32, 73,  6
280 data 240, 12,201, 32,240
290 data 247, 24,101,252,133
300 data 252, 76, 37,  6,166
310 data 252,169,  0,132,253
320 data  32, 95,164,169, 13
330 data  32,210,255,164,253
340 data  76, 17,  6,200,208
350 data   2,230, 32,177, 31
360 data  96,162,  0,189,123
370 data   6,240,  6, 32,210
380 data 255,232,208,245, 32
390 data  73,  6,170, 32, 73
400 data   6,132,253, 32, 95
410 data 164,162,  3,169, 32
420 data  32,210,255,202,208
430 data 250,169,  0,133,252
440 data 164,253, 96, 82, 69
450 data  71, 69, 76, 32,  0
460 data   0, 72,138, 72, 32
470 data 225,255,240,251,104
480 data 170,104, 96, -1

*** EINDE LISTING ***
```

De checksum van de checksum volgt volgende keer.

Tempo-typen C-16

Tempo-typen is een spel dat nogal lijkt op een onderdeel uit de Radarsoft typecursus. Als titel dus niet helemaal volgens de regels. Het programma laat een woord van rechts naar links over het beeld schuiven. Op een gegeven moment bereikt het woord een laserstraal en dreigt dan te worden vernietigd. Door de letters van het woord in te tikken vóórdat het de straal bereikt, wordt de vernietiging van het woord voorkomen en dit levert de punten op. Naarmate u beter en sneller gaat typen, wordt het spelniveau moeilijker. Een erg fraai programma dat niet in uw softwarebibliotheek zal mistaan.

```
  0 rem 1024
 10 reada$:ifa$<>"**"thend=d+1:goto10

 20 dimwo$(d)
 30 vol7
 40 color0,1:color4,1
 50 ri=1:kk=0:vol7
 60 print"[CLR-HOME][WIT][19xneer][5xrech
    ts]wilt je instructies (j/n) ?"
 70 char0,0,05,"[oranje][12xspatie]t
    empo-type"
 80 char0,0,08,"[ROOD][11xspatie]gema
    akt door"
 90 char0,0,10,"[PURPER][9xspatie]edw
    in v. oostrom"
100 char0,0,13,"[bruin][24xspatie]"
110 char0,0,13,"[GEEL][24xspatie]"
120 geta$:ifa$="n"then450
130 sound1,500+kk,3:sound2,500-kk,3:k
    k=kk+ri:ifkk=10orkk=0thenri=-ri
140 ifa$<>"j"then120
150 print"[CLR-HOME]"chr$(14)
160 print"[2xneer] WELKOM bij de eers
    te kennismaking"
170 print"[neer] van het [RVS-aan]TEM
    PO-TYPE[RVS-uit] systeem. De opze
    t"
180 print"[neer] van dit spel is om s
    pelender wijs snel"
190 print"[neer] te leren typen. Als
    je zo meteen gaat"
200 print"[neer] beginnen zal je op h
    et scherm enkele"
210 print"[neer] gegevens zien staan
    die we stuk voor"
220 print"[neer] stuk gaan behandelen
    ."
230 print"[2xneer][3xspatie][RVS-aan]
    DRUK[shift-SPATIE]OP[shift-SPATIE]
    EEN[shift-SPATIE]TOETS":getkeya$
240 print"[CLR-HOME]"
250 print"[2xneer] ten eerste : [RVS-aan]
    DE[shift-SPATIE]ENERGY-TELLER[RVS-uit
    ]. deze"
260 print"[neer] geeft aan hoeveel en
    ergy je nog "
270 print"[neer] over hebt. De hoevee
    lheid energy neemt"
280 print"[neer] af als een woord de
    lijn raakt. Als de"
290 print"[neer] teller op nul staat
    is het spel afge-"
300 print"[neer] lopen. [RVS-aan]DE[shift
    -SPATIE]SCORE[RVS-uit]-teller gee
    ft aan "
310 print"[neer] de door jouw gehaald
    e score."
320 print"[neer] [RVS-aan]DE[shift-SPATIE
    ]HI-SCORE[RVS-uit]-teller geeft d
    e tot nu"
330 print"[neer] toe hoogst gehaalde
    score.
340 print"[2xneer] [RVS-aan]DRUK[shift-SP
    ATIE]OP[shift-SPATIE]EEN[shift-SPATIE
    ]TOETS[RVS-uit]":getkeya$
350 print"[CLR-HOME][2xneer] [RVS-aan]
    DE[shift-SPATIE]LEVEL[RVS-uit]-me
    ter geeft aan op welk"
360 print"[neer] speellevel je bevind
    t. Als er 50 of"
370 print"[neer] meer tekens zijn weg
    getypt dan zal het"
380 print"[neer] speellevel worden ve
    rhoogd. Naarmate"
390 print"[neer] de speellevel toenee
    mt zal de lijn zich"
400 print"[neer] naar rechts bewegen.
     Als je begint mag"
410 print"[neer] je zelf het speellev
    el ingeven."
420 print"[neer] de [RVS-aan]TIJD[RVS-uit
    ]-meter geeft slechts een"
430 print"[neer] indicatie van hoelan
    g je speelt."
440 print"[2xneer] [RVS-aan]DRUK[shift-SP
    ATIE]OP[shift-SPATIE]EEN[shift-SPATIE
    ]TOETS[RVS-uit]":getkeya$:printchr$
    (142)
450 open2,0:print"[CLR-HOME]":char0,4
    ,8,"welke speellevel wil je[2xspatie]
    7[links]"
460 input#2,le$:le=val(le$):close2
470 ifle<=0orle=>17then450
480 le=le-1:char0,0,1,"[RVS-aan]┌CCCC
    CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
    C┐"
490 fort=2to15:char0,0,t,"B[38xrechts]
    B":next
500 char0,0,16,"├CCCCCCCCCCCCCCCCCCCC
    CCCCCCCCCCCCCCCCC┤"
510 fort=17to23:char0,0,t,"B[38xspatie]
    B":next
520 char0,0,23,"└CCCCCCCCCCCCCCCCCCCC
    CCCCCCCCCCCCCCCCC┘"
530 char0,2,17,"energy : ▒▒▒▒▒▒▒▒▒▒▒▒
    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒":en=28
540 char0,2,18,"score[2xspatie]: 0000
    000"
550 char0,2,19,"hiscore: 0000000"
560 char0,2,20,"level[2xspatie]:[6xspatie
    ]01":gosub800
570 char0,2,21,"[RVS-aan]tijd[3xspatie]
    : 0:00:00":restore
580 ti$="000000":pudef"0"
590 restore:fort=0tod:readwo$(t):next

600 a$=wo$(int(rnd(1)*d)):s$=a$
610 y1=int(rnd(1)*12)+3
620 :fort=1to38-len(a$):a$=a$+" ":next

630 fort=1tolen(a$)
640 sound1,115,1
650 char0,11,21,"[RVS-aan]":printmid$
    (ti$,2,1)":"mid$(ti$,3,2)":"right$
    (ti$,2)"[RVS-uit]"
660 fork=1to3 :r$=left$(a$,t):char1,3
    9-t,y1,r$
670 b$=mid$(a$,1,1):getc$:ifc$=b$then
    gosub710
680 ifle*2=39-tthen1020
690 ifb$=" "then720:rem score byhoude
    n
700 next:next
710 a$=right$(a$,len(a$)-1):return
720 rem score byhouden
730 te=te+len(s$):fort=1to(len(s$))
740 sc=sc+le:char0,11,18,"[RVS-aan]";
```

```
      printusing"#######";sc
 750 ifsc>hithenhi=sc
 760 char0,11,19,"[RVS-aan]":printusing
     "#######";hi
 770 sound1,1000,2:next
 780 ifte>50thente=0:gosub800:rem leve
     l hoger
 790 goto600
 800 print:le=le+1:
 810 ifle=>17thenle=le-1
 820 ifle=1then950
 830 char0,(le-1)*2-1,2,"[3xspatie]"
 840 char0,le*2-1,2,"└┬┘"
 850 fort=3to14
 860 sound1,500+t*5,5
 870 char0,(le-1)*2,t," "
 880 char0,le*2,t,"B"
 890 sound1,500-t*5,5
 900 next
 910 char0,(le-1)*2-1,15,"[3xspatie]"
 920 char0,le*2-1,15,"┌┴┐"
 930 char0,(le-1)*2-1,2,"[3xspatie]"
 940 char0,(le-1)*2-1,15,"[3xspatie]"
 950 fort=3to14
 960 char0,le*2,t,"B"
 970 next
 980 char0,le*2-1,2,"└┬┘"
 990 char0,le*2-1,15,"┌┴┐"
1000 char0,16,20,"[RVS-aan]":printright$
     ("0"+mid$(str$(le),2),2)"[RVS-uit]"

1010 return
1020 rem woord heeft lijn gepasseerd
1030 forqw=5to35
1040 r$="[RVS-aan]"+left$(a$,t):char1,
     39-t,y1,r$
1050 sound1,500-qw,5:
1060 r$="[RVS-uit]"+left$(a$,t):char1,
     39-t,y1,r$
1070 sound1,500+qw,5:
1080 next
1090 fora=1tolen(a$)
1100 ifmid$(a$,a,1)="[shift-SPATIE]"then
     1120
1110 next
1120 forqw=1tolen(s$)
1130 ifpeek(3762+en)=102+128thenpoke37
     62+en,92+128:goto1170
1140 sound3,en*10,5
1150 ifpeek(3762+en)=92+128thenpoke376
     2+en,160:en=en-1:poke3762+en,102+
     128
1160 ifen=1then1210
1170 next
1180 char0,39-t,y1,"[ZWART]":printleft$
     (a$,t)"[WIT]"
1190 char0,39-t,y1,"[WIT]B"
1200 goto600
1210 char0,8,7,"[RVS-aan]je hebt geen
      energy meer.[RVS-uit]"
1220 char0,1,9,"wil je het nog een kee
     r proberen (j/n)"
1230 ri=1:kk=0
1240 geta$:ifa$<>""then1240
1250 geta$:ifa$="j"thente=0:goto1290
1260 sound1,500+kk,3:sound1,500-kk,3:k
     k=kk+ri:ifkk=100orkk=0thenri=-ri
1270 ifa$<>"n"then1250
1280 print"[CLR-HOME] tot ziens....":end

1290 le=0:sc=0:
1300 goto30
1310 rem data's volgen nu
1320 datahallo,cassette,boek,computer,
     stoel,buro,pen,potlood,band,plaat
     ,asbak,bed
1330 datakussen,auto,wielen,muziek,raa
     m,gordijn,verwarming,tijdschrift
```

```
1340 datalamp,strijkijzer,kapstok,werk
     en,surfplank,vakantie,boekenkast,
     tafel,huisdier
1350 datamatteklopper,ijslollie,koffie
     ,thee,schoolbord,pretpark,slaapmu
     ts,gehaktbal
1360 datatas,ijsmuts,bezien,waardig,la
     mpenkap,kattebak,schoonmaken,kett
     ing,broek
1370 datakoptelefoon,telefoon,muur,lad
     e,bijl
1380 dataonschuldig,maatschappij,verja
     ardag,schilderij,musea,kat,schoen
     en,pistool
1390 datasherry,fles,ledematen,biljart
     en,schoonheid,wedstrijd,disketten
     ,sigaret
1400 datahuiswerk,dropjes,horizon,rech
     tbank
1410 dataprullebak,zonnebank,bewindsma
     n,minister,glimlach,draagvlak,ver
     zorgingsgebied
1420 datadienst,regeling,knorrig,moord
     dadig,tederheid,vrolijkheid,slaap
     zak,doorzichtig
1430 dataroekeloos,trap,kantine,lokaal
     ,lessenaar,schemering,wegstromen,
     bitter
1440 datapersoonlijk,bakoven,zwaluw
1450 dataonmiddellijk,klerenkast,reken
     machine,deur,stekker,veer,agenda,
     afschuw
1460 datawereld,aanwezig,hoeveelheid,o
     nderwerp,kapitaal,veroveraar,hoek
     igheid
1470 datasleutel,toilet,wasbak,zeep,sc
     heerschuim,geld,kaptafel,telefoon
     ,bril
1480 dataaluminium,folie,kabel,televis
     ie,video,tuner,beurs,veiling,prin
     ter
1490 dataverbazing,kleding,woedend,luc
     hthaven,bromfiets,rationaal
1500 datavoorziening,verkeer,stoplicht
     ,regionaal,politiek,concurrentie,
     belangstelling
1510 datawolkenkrabber,hoofdbrekers,st
     uur,glas
1520 datahandeling,transport,richting,
     grondstof,fabriek,wegen,oordeel,v
     erschil
1530 datagroter,aanwezig,ellendeling,s
     tructuur,mogelijk,wanneer,vervang
     ing
1540 datafusie,melkweg, kartel, centra
     al,minachting,waarde,,weinig,gesc
     hiedenis,recht
1550 datainitiatief,molecuul,president
     ,resevoir,referdum,nationaal,cong
     res
1560 datameineed,produkt,rekening,risi
     co,voortbestaan,uitgeven,gemakkel
     ijk,vakbond
1570 dataaandeel,werknemers,terminals,
     mobiel,mechanisering,revolutie,en
     ergie
1580 datarecorder,multimap,landkaart,f
     loder,kanon,reactor,bierglas,papi
     er,behang
1590 dataliim,quotient,dimmer,explosie
     ,expositie,recent,onbeschrijfelij
     k,teneinde
1600 dataonderzoek,educatie,duizelig,s
     tation,plein,trechter,balans,waar
     achtig
1610 datavuil,verbranding,hanteerbaar,
     etage,garage,verdieping,exclusief
```

```
      ,druiloor
1620  datadroom,bereizen,schaduw,fototo
      estel,filmrol,aansturen,naarmate,
      jager
1630  datamuizen,quarantaine,dokter,pos
      tbode,nestelen,nikkel,gulden
1640  datanoordzee,nonsens,noodgedwonge
      n,schuiftafel,reglement,taalkunde

1650  data**
1660  rem
1670  rem tempo typen / c16 & plus4
1680  rem door edwin van oostrom
1690  rem uit schiedam
1700  rem
1710  rem 1024

*** EINDE LISTING ***
```

```
tempo-typen c16

regel 0    86     regel 540   121
regel 10   242    regel 550   11
regel 20   229    regel 560   222
regel 30   18     regel 570   14
regel 40   38     regel 580   98
regel 50   124    regel 590   81
regel 60   204    regel 600   45
regel 70   232    regel 610   107
regel 80   186    regel 620   133
regel 90   51     regel 630   213
regel 100  1      regel 640   43
regel 110  10     regel 650   189
regel 120  180    regel 660   32
regel 130  63     regel 670   164
regel 140  28     regel 680   129
regel 150  237    regel 690   202
regel 160  115    regel 700   62
regel 170  191    regel 710   223
regel 180  49     regel 720   105
regel 190  46     regel 730   139
regel 200  160    regel 740   224
regel 210  234    regel 750   227
regel 220  64     regel 760   137
regel 230  102    regel 770   18
regel 240  112    regel 780   113
regel 250  202    regel 790   31
regel 260  110    regel 800   188
regel 270  233    regel 810   62
regel 280  248    regel 820   68
regel 290  178    regel 830   130
regel 300  147    regel 840   113
regel 310  197    regel 850   195
regel 320  248    regel 860   12
regel 330  93     regel 870   200
regel 340  248    regel 880   93
regel 350  14     regel 890   13
regel 360  153    regel 900   130
regel 370  119    regel 910   182
regel 380  3      regel 920   152
regel 390  148    regel 930   130
regel 400  129    regel 940   182
regel 410  186    regel 950   195
regel 420  73     regel 960   93
regel 430  229    regel 970   130
regel 440  122    regel 980   113
regel 450  13     regel 990   152
regel 460  235    regel 1000  17
regel 470  253    regel 1010  142
regel 480  133    regel 1020  150
regel 490  231    regel 1030  28
regel 500  191    regel 1040  96
regel 510  206    regel 1050  186
regel 520  201    regel 1060  224
regel 530  135    regel 1070  185
```

```
regel 1080   130    regel 1450   183
regel 1090   194    regel 1460   145
regel 1100   214    regel 1470   43
regel 1110   130    regel 1480   196
regel 1120   59     regel 1490   67
regel 1130   236    regel 1500   66
regel 1140   58     regel 1510   24
regel 1150   241    regel 1520   240
regel 1160   108    regel 1530   17
regel 1170   130    regel 1540   222
regel 1180   119    regel 1550   246
regel 1190   148    regel 1560   22
regel 1200   31     regel 1570   90
regel 1210   227    regel 1580   38
regel 1220   246    regel 1590   230
regel 1230   48     regel 1600   144
regel 1240   70     regel 1610   88
regel 1250   33     regel 1620   107
regel 1260   110    regel 1630   160
regel 1270   85     regel 1640   33
regel 1280   98     regel 1650   215
regel 1290   95     regel 1660   143
regel 1300   236    regel 1670   27
regel 1310   145    regel 1680   82
regel 1320   58     regel 1690   191
regel 1330   208    regel 1700   143
regel 1340   10     regel 1710   86
regel 1350   101
regel 1360   85
regel 1370   85     ready.
regel 1380   172
regel 1390   54
regel 1400   75
regel 1410   244
regel 1420   39
regel 1430   183
regel 1440   27
```

Morse C-16

Een eenvoudig programma om morse signalen te maken. Een ingetikte letter wordt direkt door de computer in de bekende fluitsignalen omgezet. Het is niet echt gemakkelijk om de signalen te lezen. Men zal dan ook wel even moeten oefenen alvorens men de pieptonen kan terugfluiten. Overigens een leuke toepassing van de C-16 met zijn relatief eenvoudige sound-commando's.

```
  1 rem morse / c16 & plus4
  2 rem door stefan wieman
  3 rem uit goirle
  4 rem 1026
  5 rem
 90 vol8:print"[CLR-HOME][5xneer][7xrecht
    s]letter?[2xneer]"
100 getkeya$:ifa$="" "then100
110 a=asc(a$)-64
120 ifa<1ora>26then130:else160
130 print"[CLR-HOME][6xneer][5xspatie]
    doe alleen letters"
140 forte=1to1000:next
150 print"[CLR-HOME]":goto100
160 ifa>10anda<21then190
170 ifa>20then200
180 onagoto210,230,250,270,290,310,33
    0,350,370,390
```

```
190 a=a-10:onagoto410,430,450,470,490
    ,510,530,550,570,590
200 a=a-20:onagoto610,630,650,670,690
    ,710
210 b$=". C"
220 t1=1:t2=2:goto730
230 b$="C . . ."
240 t1=2:t2=1:t3=1:t4=1:goto730
250 b$="C . C ."
260 t1=2:t2=1:t3=2:t4=1:goto730
270 b$="C . ."
280 t1=2:t2=1:t3=1:goto730
290 b$="."
300 t1=1:goto730
310 b$=". . C ."
320 t1=1:t2=1:t3=2:t4=1:goto730
330 b$="C C ."
340 t1=2:t2=2:t3=1:goto730
350 b$=". . . ."
360 t1=1:t2=1:t3=1:t4=1:goto730
370 b$=". ."
380 t1=1:t2=1:goto730
390 b$=". C C[shift-SPATIE]C"
400 t1=1:t2=2:t3=2:t4=2:goto730
410 b$="C . C"
420 t1=2:t2=1:t3=2:goto730
430 b$=". C[shift-SPATIE]. ."
440 t1=1:t2=2:t3=1:t4=1:goto730
450 b$="C[shift-SPATIE]C"
460 t1=2:t2=2:goto730
470 b$="C ."
480 t1=2:t2=1:goto730
490 b$="C[shift-SPATIE]C[shift-SPATIE]
    C"
500 t1=2:t2=2:t3=2:goto730
510 b$=". C[shift-SPATIE]C[shift-SPATIE]
    ."
520 t1=1:t2=2:t3=2:t4=1:goto730
530 b$="C[shift-SPATIE]C[shift-SPATIE]
    . C"
540 t1=2:t2=2:t3=1:t4=2:goto730
550 b$=". C ."
560 t1=1:t2=2:t3=1:goto730
570 b$=". . ."
580 t1=1:t2=1:t3=1:goto730
590 b$="C"
600 t1=2:goto730
610 b$=". . C"
620 t1=1:t2=1:t3=2:goto730
630 b$=". . . C"
640 t1=1:t2=1:t3=1:t4=2:goto730
650 b$=". C C"
660 t1=1:t2=2:t3=2:goto730
670 b$="C . . C"
680 t1=2:t2=1:t2=1:t3=2:goto730
690 b$="C . C C"
700 t1=2:t2=1:t3=2:t4=3:goto730
710 b$="C[shift-SPATIE]C[shift-SPATIE]
    . ."
720 t1=2:t2=3:t3=1:t4=1:goto730
730 print"[2xspatie]";a$;" = ";b$:print

740 t1=t1*10:t2=t2*10:t3=t3*10:t4=t4*
    10
750 vol8:sound1,900,t1:sound1,1017,5:
    sound1,900,t2:sound1,1017,5
760 :    sound1,900,t3:sound1,1017,5:
    sound1,900,t4:sound1,1017,5
770 forte=1to100:nextte:clr:goto100

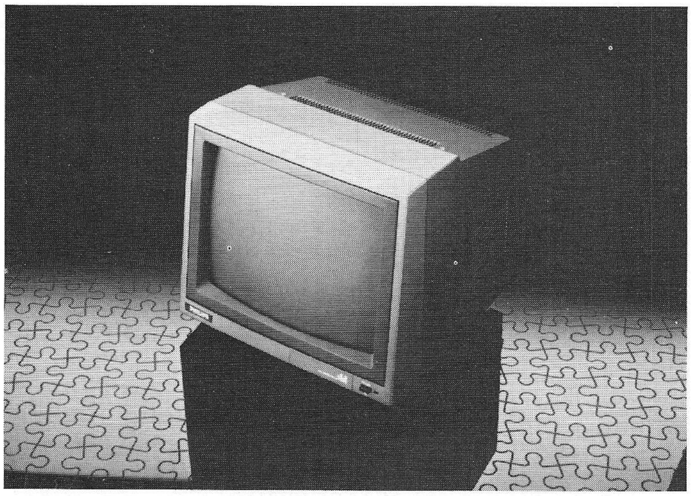

*** EINDE LISTING ***
```

morse c16

regel 1	140	regel 420	14
regel 2	69	regel 430	73
regel 3	67	regel 440	178
regel 4	88	regel 450	130
regel 5	143	regel 460	106
regel 90	14	regel 470	77
regel 100	87	regel 480	105
regel 110	132	regel 490	229
regel 120	155	regel 500	15
regel 130	130	regel 510	126
regel 140	30	regel 520	179
regel 150	196	regel 530	19
regel 160	37	regel 540	180
regel 170	24	regel 550	123
regel 180	210	regel 560	13
regel 190	96	regel 570	230
regel 200	81	regel 580	12
regel 210	77	regel 590	31
regel 220	105	regel 600	198
regel 230	169	regel 610	123
regel 240	178	regel 620	13
regel 250	62	regel 630	169
regel 260	179	regel 640	178
regel 270	123	regel 650	16
regel 280	13	regel 660	14
regel 290	138	regel 670	62
regel 300	197	regel 680	177
regel 310	169	regel 690	211
regel 320	178	regel 700	181
regel 330	16	regel 710	126
regel 340	14	regel 720	180
regel 350	20	regel 730	173
regel 360	177	regel 740	222
regel 370	184	regel 750	192
regel 380	104	regel 760	177
regel 390	115	regel 770	177
regel 400	180		
regel 410	16	ready.	

Een van de sterke punten van de CBM-128 is zijn mogelijkheid om 80 karakters per regel weer te geven. Dat betekent een resolutie van 640 x 200 punten. Het is dan ook heel jammer dat Commodore er niet voor gezorgd heeft dat de bit-map instructies voor het 40-koloms scherm ook in de 80-koloms mode werken. Dit is echter betrekkelijk eenvoudig zelf te verwezenlijken, en Peter Heinckiens gaat in op de basistechnieken daarvoor.

Hoge Resolutie op het 80-koloms scherm van de C-128



door Peter Heinckiens

De beeldresolutie, dus het aantal aan te sturen punten en daarmee de mate van detaillering van het beeld is bij de 128 tamelijk goed. De VDC-videochip doet het heel goed en vormt eigenlijk de basis van de "serieuze" toepassing.

Wanneer we nu die VDC-chip zodanig willen gaan aanspreken, dat we wat van de bekende C-64 effecten ook op 80 koloms kunnen bereiken, moeten we dus weten hoe die chip valt te adresseren, en welke commando's we kunnen geven. Dat is helaas geen zaak, die in Basic valt op te lossen, hier gaan we op de machinetaal-toer.

Adresseren

Een van de eerste verschillen tussen de VDC-chip (de chip die voor het 80-koloms beeldscherm zorgt) en de welbekende VIC, is de manier van adresseren. Waar men bij de VIC rechtstreeks het betreffende register kan aanspreken, moet dit bij de VDC indirect gebeuren. Dit doet men door middel van de adressen $D600 en $D601. In $ D600 plaatst men het nummer van het register dat men wil aanspreken (bits 0 tot 5). Met behulp van deze geheugenplaats kan men ook onderzoeken of de processor reeds klaar is met zijn laatste opdracht: zo ja, dan is bit 7 nul en anders is deze bit "geset".
Het adres $D601 wordt gebruikt om de data door te geven: via dit adres leest men of schrijft men naar het register.
Dit alles vertaald naar een programma geeft ons de volgende twee routines:

```
PLAATS STY $D600 ; registernummer in $D600
LOOP  BIT $D600 ; test bit 7
      BPL LOOP ; indien geset: nog niet klaar
      STA $0601 ; plaats de data in $D601
      RTS
LEES STY $D600
LOOP BIT $D600
BPL LOOP
LDA $D601 ; lees de data in $D601
RTS
```

De bitmapmode

De bitmapmode van de VDC verschilt wel enigszins van die van de VIC : het geheugen is hier namelijk niet in rijen van 8 verdeeld, met als gevolg dat het veel gemakkelijker te adresseren is. Verder zit dit 16 KB geheugenstuk in een RAM, die speciaal voor de VDC gereserveerd is, en enkel toegankelijk is via indirecte adressering: in register 18 en 19 plaatst men het adres dat men wil aanspreken (HI/LO), en de data komt in register 31.

Het aanschakelen.

Het inschakelen van de bitmapmode gebeurt d.m.v. register 25: bit 7 (TEXT) daarin moet "geset" worden en ook moet bit 6 (ATR) "gecleard" worden. Deze laatste zorgt er voor, dat alle karakters monochroom weergegeven worden. Dit is weer nodig omdat het kleur-geheugen overschreven wordt door het HiRes-scherm. Deze twee operaties kunnen gecombineerd worden, door in register 25 de waarde 128 te plaatsen.

```
AAN LDY #$ 19 ; register 25
    LDA #$ 80 ; met 128 laden
    JSR PLAATS
    RTS
```

Het beeldscherm wissen

Daar het geheugen bijna 16K groot is (640/8 * 200 bytes om precies te zijn), moeten we dus 16K bytes nul maken. Dit gebeurt d.m.v. het volgende programma:

```
WISSEN LDX #$ 40 ; 16K = 64 * 256 BYTES
LOOP   LDY #$ 12 ; register 18 (update HI)
       TXA       ; plaats HI-byte
       JSR PLAATS
       LDY #$1F ; data-register
       LDA #$00 ; adres met nul vullen
       JSR PLAATS
```

```
DEY      ; wordcount-register
JSR PLAATS ; vullen met $00
DEX      ; HI : = HI-1
BPL LOOP ; volgende 256 Bytes
RTS
```

Indien u de bovenstaande routine nauwkeurig bekeken heeft, zult u zich ongetwijfeld afgevraagd hebben of het niet nodig is om ook de LO-byte van het update-adres aan te passen. Dit wordt echter vermeden door gebruik te maken van het wordcount-register : door dit met 0 te laden, wordt een heel blok van 256 bytes ineens gewist.
Deze methode heeft wel een nadeel : soms wordt niet heel het scherm gewist. Dit is echter makkelijk op te vangen : roep de routine twee keer aan.

```
START JSR WISSEN ; roep eerste maal aan
      JMP WISSEN ; roep tweede maal aan
```

Uitschakelen

Om de bitmapmode uit te schakelen, moet de TEXT-bit gecleard en de ATR-bit terug geset worden. Ook moet de karakterrom (CHARROM) terug gecopieerd worden (d.m.v. de routine op $CE0C). We krijgen dus:

```
UIT  LDY #$19 ; register 25
     LDA #$40 ; laden met 64
     JSR PLAATS
     JMP $CE0C ; kopieer CHARROM
```

Een pixel aanschakelen

En dan komen we nu aan het belangrijkste deel van ons programma: Het aanschakelen van een welbepaald punt. Zoals reeds gezegd, is bij de VDC het schermgeheugen niet in rijen van acht verdeeld, zodat de correcte byte gevonden wordt door de volgende formule:

BY = INT (X/8) + Y*80

Hoe vinden we dit nu? Daar we per byte acht pixels kunnen controleren, vinden we dus de correcte byte door de x-coördinaat te delen door acht, en het resultaat naar beneden af te ronden. Bijgevolg vinden we dat er per regel 640/8 = 80 bytes nodig zijn. We vinden de correcte blok bytes door de Y-coördinaat met 80 te vermenigvuldigen, wat ons dan bovenstaande formule geeft. De geheugenplaatsen in onderstaande routines worden als volgt benut:

eerste/tweede : X-coördinaat LO/HI
derde : Y-coördinaat

a. Bereken INT(X/8) (resultaat in eerste/tweede)

```
PHA ; X-LO in stack
LSR tweede ; deel HI-byte door 2
ROR eerste ; en plaats carry in LO-byte
LSR tweede
ROR eerste
LSR tweede
ROR eerste
```

b. Bereken Y*80 (resultaat in accu/vierde)

```
      CLC          ; carry wissen
      LDA #$00     ; eerst berekenen we Y*5
      STA vierde   ; HI-byte van Y*5 is nog nul
      LDA derde    ; Y naar accu
      LDY #$04     ; daar 4 maal Y
LOOP  ADC derde    ; bij optellen
      BCC CONT     ; indien nodig HI-byte
      INC vierde   ; met 1 verhogen
      CLC          ; carry wissen
CONT  DEY          ; teller verlagen
      BNE LOOP     ; indien niet klaar : LOOP
      ASL          ; nu vermenigvuldigen we het
      ROL vierde   ; resultaat met 2↑4 = 16
      ASL
      ROL vierde
      ASL
      ROL vierde
      ASL
      ROL vierde
```

c. Bereken BY (resultaat in derde/vierde)

```
      ADC eerste   ; accu + LO van INT(X/8)
      STA derde    ; resultaat in derde
      LDA tweede   ; HI van INT(X/8)
      ADC vierde   ; + HI van Y*80 (+carry)
      STA vierde   ; resultaat in vierde
```

Nu dienen we in die byte nog enkel de correcte bit te berekenen, en dit gebeurt door middel van de volgende formule:
BI = 7 - (X AND 7)
Immers: we vinden de gezochte bit door de rest van de deling van X door 8 te beschouwen:
X - INT (X/8)*8

Nu beginnen we de bits echter van rechts te tellen:
7 6 5 4 3 2 1 0

dus wordt de formule:

BI = 7 - (X - INT(X/8)*8)

De laatste term hierin vraagt redelijk veel rekenwerk, en doet dus de vraag rijzen, of er geen eenvoudiger methode bestaat om de rest van de deling te vinden. Die bestaat inderdaad: voer een logische "AND" uit tussen X en 7. Stel dat we bijvoorbeeld de rest van 30/8 zoeken, dan krijgen we:

```
      0 0 0 1 1 1 1 0        (30)
AND   0 0 0 0 0 1 1 1        (7)
---------------------
      0 0 0 0 0 1 1 0        (6)
```

hetgeen ons dus inderdaad 6 oplevert. Bijgevolg wordt de uiteindelijke formule:
BI = 7 - (X AND 7)

Dit levert ons het volgende programma op:

```
BI    PLA          ; haal X-LO van stack
      SEC          ; set carry
      AND #$07     ; X AND 7
      STA eerste   ; Plaats resultaat in eerste
      LDA #$07
      SBC eerste   ; 7 - (X AND 7)
      TAY          ; resultaat in Y
```

Nu hebben we dus de byte en het bitnummer gevonden, en dienen we nog enkel die betreffende bit te "setten". In BASIC zou dit als volgt gebeuren:

POKE BY, PEEK(BY) OR 21BI

Dit alles vertaald naar machinetaal geeft ons:

a. Bereken 21BI (resultaat in eerste stukje listing)

```
2↑BI  LDA #$01     ; 2↑BI is minstens 1
      CPY #$00     ; Y = 0 ?
      BEQ CONT     ; ja : Klaar
LOOP  ASL          ; accu maal 2
```

```
        DEY              ; verminder teller
        BNE LOOP         ; Y<>0 : doe nogmaals x2.
CONT    STA eerste       ; resultaat in eerste
```

b. Voer de POKE uit

```
POKE    LDY #$12         ; UPDATE - HI
        LDA vierde       ; instellen met vierde
        JSR PLAATS
        INY              ; UPDATE - LO
        LDA derde        ; instellen met derde
        JSR PLAATS
        LDY #$1F         ; haal de waarde uit het
        JSR LEES         ; DATA - register
        ORA eerste       ; OR met 2†BJ
        PHA              ; bewaar dit
        LDY #$12         ; zie hoger
        LDA vierde
        JSR PLAATS
        INY
        LDA derde
        JSR PLAATS
        PLA              ; haal de nieuwe waarde
                           van stack
        LDY #$1F         ; Plaats die in het adres
                           bepaald
        JSR PLAATS       ; door derde/vierde
        RTS
```

Het programma

De bovenstaande routines vindt u hieronder samengevoegd tot een programma. De assembler-listing kunt u direct met de 128-monitor intypen. Daar sommigen dit prefereren, heb ik ook een BASIC-lader bijgevoegd. Denk erom dat u het programma wegschrijft voordat u het runt. De machinecode alleen schrijft u als volgt weg:

BSAVE "MINIGRAPH - M.C", P0B00 TO P0BC5

Listing 3 is een demo-programma: dit tekent een sinusoide. U kunt vanzelfsprekend ook een andere functie ingeven.

Tenslotte nog een programma, dat een aantal hierboven behandelde za e*Z*lustreert. Het programma MINIGRAPH tekent een sinusoide op het scherm, maar je kunt zelf natuurlijk andere functies gaan ingeven. Voor het werken met dit programma zijn een aantal SYS commando's nodig.

Het aanschakelen en wissen van de bitmap gaat met SYS DEC("0B18").

Het uitschakelen van de bitmap gaat met SYS DEC("0B3A")

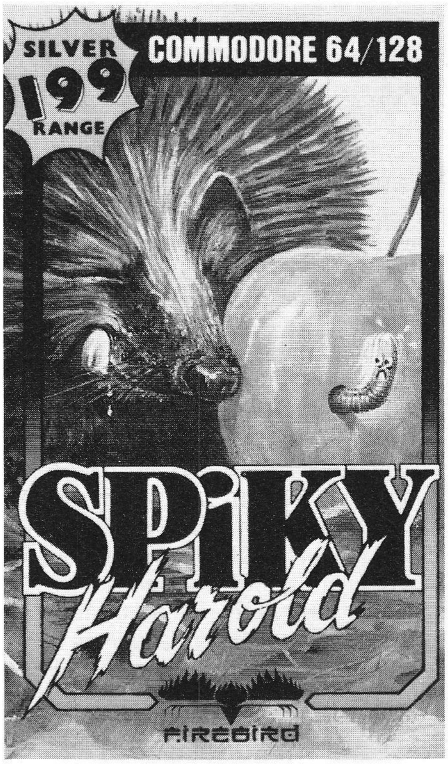
Het plotten van een punt met SYS DEC("0B49"),X-LO,X-HI,Y

X-LO en X-HI worden als volgt berekend:

X-LO= X and 255   X-HI = X/256

Spiky Harold

Dit spel, ook van Firebird, is alweer niet veel soeps. Het lijkt er op dat er na Elite, het grote succes van dit softwarehouse, alleen nog maar waterige aftreksels van oudere spellen gemaakt kunnen worden. Ook bij dit programma zijn de graphics tweedimensionaal en weinig geanimeerd, de muziek is matig en het spelidee (ook dit spel speelt in een doolhof) is niet origineel. Al met al kunt u dit programma beter niet aanschaffen. ●



Countdown to Meltdown

Dit Mastertronicspel behandelt een tegenwoordig helaas populair onderwerp. Al weer een ongeval in een kerncentrale en de reactorkern dreigt te smelten. Leidt uw 8 robots door de ruim 2000 kamers op 8 verschillende niveaus (de constructeurs van de centrale waren blijkbaar niet goed wijs, vandaar natuurlijk het ongeval) naar de kern. Daar dienen de remstaven gevierd te worden. Tevens zitten er nog wat lastige indringers in het gebouw. Laat de robots hen tijdig uitschakelen voordat zij zelf het slachtoffer worden.
Een boeiend C-64/C-128 spel met matige 3D graphics voor ca. *f* 12,- ●

Scarab

Scarab is een beeldadventure van het softwarehouse Geposoft. Een beeld- of actie-adventure is een programma waarbij er niet, zoals bij een 'normale' adventure met behulp van tekst gecommuniceerd wordt. Met de joystick wordt te kennen gegeven wat er gebeuren moet.

Het verhaal van Scarab speelt in het oude Egypte. De hoofdrol in dit spel heeft de ongelukkige, die door een gemene priester van de farao Amenhotb III in een scarabee (mestkever) is veranderd. Om het verhaal helemaal compleet te maken is deze scarabee opgesloten in een pyramide, en het is de bedoeling om eruit te komen. Dan zal hij weer aan het normale leven kunnen deelnemen. U begrijpt het al, dit is een doolhofspel in een pyramide.

De kwaliteit van de "graphics" vinden we vrij matig, want de plaatjes zijn, afgezien van de laadplaat, tweedimensionaal en niet zo goed geanimeerd. Het spelidee, een doolhofadventure, is en blijft goed, niet erg origineel, maar het ontsnappen uit een doolhof is een van de betere computer uitdagingen.

Al met al is het een redelijk leuk spel, maar we vrezen dat de ervaren speler er vrij gauw op uitgekeken zal zijn.

Caverns of Xydrapur

Ook van Geposoft is het spel 'Caverns of Xydrapur'. Dit programma is alweer een actie-adventure maar hier een wat afgezaagd spelidee. Alweer moet men z'n weg vinden door een grottenstelsel maar nu met het doel de mensheid te redden van de gemene onderaardse beschaving in de grotten.

De uitvoering is niet best: weinig kleur, tweedimensionaal en niet geanimeerd zijn de beschrijvingen die voor de meeste platen passen. ●

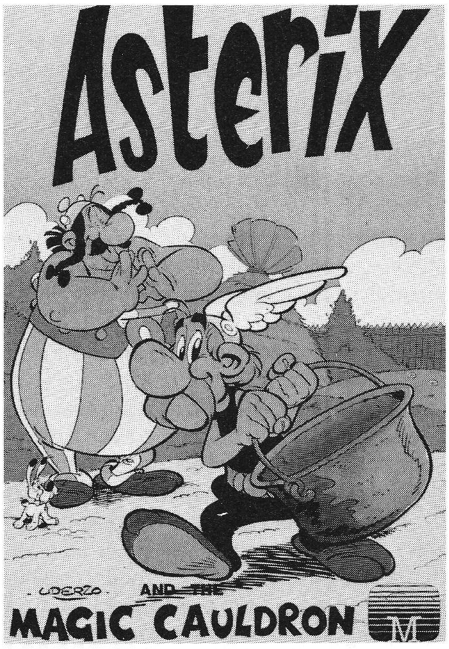
Asterix and the magic Cauldron

Dit fraaie programma werd ontwikkeld door Melbourne House. Dit softwarehuis is vooral beroemd om de grandioze Oosterse vechtsport spellen die het op de makt brengt.

Bij Asterix, de bekende stripheld uit de romeinse tijd, wordt ook aardig gevochten. De kracht, die de dappere galliërs tot nu toe behoedde voor overheersing door de romeinen komt niet van Oosterse wijsheid of kruiden, maar wordt ontleend aan de bekende toverdrank die bereid wordt door de druide. Er dreigt echter iets mis te gaan, want de magische ketel voor de superdrank is gebroken en de stukken zijn weggeraakt. Aan Asterix is de taak om de stukken bij elkaar te zoeken en de zaak tot een goed einde te brengen. Hij wordt hier ook bijgestaan door Obelix, als er maar genoeg everzwijn voorhanden is......

Het spel is dus een zoekspel, een actie-adventure, met mooie, geanimeerde graphics. Behalve het adventuredeel kent het programma ook een "fight mode", waarin de Romeinen die Asterix tegenkomt, in elkaar geslagen dienen te worden. In die mode worden de figuurtjes wat groter weergegeven, zodat het dan een echt vechtspel is.

Het op zoek gaan naar de stukken van de ketel gebeurt in een fraaie driedimensionale omgeving, de goed herkenbare figuren kunnen daar hun gang gaan. De "plattegrond" van de adventure is behoorlijk groot: de gebeurtenissen spelen zich natuurlijk af in het dorp en de omliggende romeinse legerplaatsen, maar het is zelfs mogenlijk om bijv. Rome te bezoeken.



Voor de Asterix en Obelix-liefhebbers is dit spel een 'must', maar we denken dat de kwaliteit van het programma ook anderen wel zal bevallen. ●

Basicode-3

Nieuwe kans voor programmeertolk

Er is een nieuwe versie van Basicode. In overeenstemming met de traditie krijgt deze nieuwe en geheel verbeterde computertolk het volgnummer drie mee. De eerste twee versies zijn alweer vergeten. Dat wil zeggen, er is wat onenigheid geweest tussen NOS en uitgeverij Kluwer over deze versie.

Wanneer hobbyisme en commercie samen gaan, komen er wel eens meer problemen. Basicode-3 heeft in ieder geval heel wat meer te bieden dan zijn beide voorgangers samen. Voor diegenen die nog niet weten waar het over gaat: Basicode-3 is een tolk tussen verschillende computersoorten. Het maakt het mogelijk om volgens de Basicode-3 voorschriften gemaakte programma's in elke deelnemende computer te laden en te runnen.

Een geheel andere aanpak voor Basicode, deze keer. De oudere Basicodes moesten worden besteld via de NOS door overmaking van een bepaald bedrag. Het bandje werd dan thuis gestuurd. Er werd een groot boek meegestuurd met daarin de eigenschappen van Basicode en handleidingen voor alle betrokken computersoorten. Je zult zeggen, alleen voor insiders dus. Dat was natuurlijk niet echt zo, maar het leek er wel beetje op. Basicode is daardoor waarschijnlijk nooit werkelijk aangeslagen bij het grotere programmerende publiek. Jammer eigenlijk.

Uitgever

Deze keer gaat dat geheel anders lopen. Althans het gaat er op lijken, dat de makers en medewerkers van Basicode (nu in een stichting ondergebracht) er deze keer wat meer uit willen slepen. Het cassettebandje draagt het Kluwer-embleem en ook het bijbehorende boekwerkje is door Kluwer uitgegeven. Nu is zo'n boek natuurlijk in goede handen bij een uitgeverij met een goede reputatie op het gebied van technische boeken. De markt op met Basicode-3 lijkt het devies. Het is heel begrijpelijk, dat de makers van Basicode eindelijk eens een andere honorering willen ontvangen dan een pluim en een schouderklop, alleen is het dan de vraag of het Hobbyscoop radioprogramma dan niet in de commerciële knel komt. Maar aan de andere kant: Basicode-3 is een volwassen stuk programmatuur en mag best uit de hobbyistische sfeer.

Een tal van computers doen mee: een aantal Acorn modellen (o.a BBC en Electron), Apple, alle MSX-1 en -2 computers, de oude getrouwe Philips P2000 T en M, de Sinclair Spectrum, twee Spectravideo modellen en natuurlijk niet als allerlaatste de Commodore 64. De VIC-20 de PET en de CBM 3000, 4000 en 8000 series zijn helaas afgevallen. Die moeten het nog doen met versie 2 van Basicode. Van de andere merken zijn er ook een paar verdwenen en er zijn enkele bijgekomen.

Protocol

Wie zich met Basicode gaat bezighouden, zal zich moeten houden aan een bepaald protocol. Bepaalde Basic sleutelwoorden mogen niet worden gebruikt. Bepaalde handelingen mogen niet rechtsstreeks worden uitgevoerd maar moeten lopen via de vertaalroutines van Basicode. Het wissen van het scherm gaat immers op elke computer weer anders. De vertaalroutines liggen vast in de subroutines van de vertaler. Wie in Basicode het scherm wil wissen programmeert dat met: GOSUB 100. Met GOSUB 210 wacht de computer tot er een toets wordt ingedrukt, enzovoort. Zo is er voor iedere handeling, die heel computerspecifiek is een subroutine. Gezien de primitieve Basic van de C-64 zou je Basicode-3 wel kunnen beschouen als een Basic-uitbreiding! Veel routines maken het programmeren op de C-64 eenvoudiger, zoals de mogelijkheid om de cursor te positioneren, in hires een lijn te trekken, het schakelen van en naar grafische schermen, openen en opslaan van files, het aansturen van de printer en nog vel meer. Alleen de muzikale kwaliteiten van de C-64 worden geen eer aangedaan. Maar ja, Basicode blijft een soort grote gemene deler en geen van de andere deelnemende computers is zo muzikaal ingesteld als de C-64. Tot slot, en dat is het belangrijkste, schrijft Basicode-3 het programma op een bepaalde manier naar de cassette. Dat gebeurt net zo als bij alle andere deelnemende computersoorten. Daarin ligt ook het geheim van de uitwisselbaarheid.

Boek-pak

Het gehele pakketje bestaat dus uit een

TROS

Basicode 3 programma's wekelijks bij TROS radio

Een proef met het uitzenden van basicode 3 programma's door de TROS in de afgelopen zomer heeft zoveel positieve reakties opgeleverd, dat de programmaleiding van TROS Radio besloten heeft op Radio 5 zendtijd vrij te maken voor een wekelijkse uitzending van deze programma's.

Als vaste tijd op Radio 5 is gekozen voor 17.41 uur. Basicode 3 staat zo geprogrammeerd aansluitend aan de Tros huiswerklijn.

Het programma omvat steeds het computerbulletin en een nieuw computerprogramma in basicode 3, het 'Experanto' voor thuiscomputers.

De wekelijkse uitzendingen worden geproduceerd in samenwerking met de Stichting Basicode 3.

boek en een cassette. Niet op disk, omdat de cassette natuurlijk het medium is, waar alle computers mee kunnen werken. En dat is een voorwaarde voor de uitwisseling. Er zijn immers zoveel diskformaten, dat zou niet zinnig zijn. Wel kunnen op de C-64 bestanden naar disk worden geschreven, maar geen programma's. Op de cassette staan de vertaalprogramma's voor alle computersoorten, gewoon in de standaardcodering van die bewuste computer. Na het inladen van de vertaler kan men met programmeren in Basicode-3 beginnen. Op de cassette staan tevens drie demonstratie-programma's , een grafisch programma, een muziekje en een adressenbestandsprogramma compleet met bestand (alle adressen van gebruikersverenigingen staan daar in). Ze geven een aardige indruk waartoe Basicode-3 instaat is.

Het boekje ziet er aantrekkelijk uit en bevat alle informatie voor de gebruiker. Zelfs een beginnende computergebruiker kan ermee uit de voeten. Het boek doet omvangrijk aan, maar het valt tegen: immers alle deelnemende computersoorten staan er uitgebreid in. Een C-64 gebruiker zal niet direkt het Apple-deel van het boek door gaan nemen. Maar gezien de doelstelling van het geheel is dit natuurlijk niet te vermijden. Ook lijkt het, of de auteurs van het boek (het zijn er vele) niet van elkaar wisten wat ze schreven. Veel verhalen worden dubbel gedaan en ook andere informatie, hoe belangrijk ze ook moge zijn, wordt meerder malen uitgebreid afgedrukt. Soms woordelijk identiek. Was de eindredakteur even wat afwezig, of is dit met opzet gebeurd. Als dat laatste zo was, onderschat de redakteur in kwestie de intelligentie van de computergebruiker ons inziens schromelijk. Een beetje computerhobbyist verdient meer respect! Erg goed en prima gedocumenteerd is het hoofdstuk over de voorbeeldprogramma's. In de bijlagen staat ook voldoende interessante dingen te lezen. Al met al is het Basicode-3 duo (boek en cassette) zeer de moeite waard.

Vertaalprogramma

Waar liggen nu de toepassingsmogelijkheden van een dergelijk vertaalprogramma. Dat ligt ogenschijnlijk voor de hand, maar als je wat verder doordenkt..... Behalve dat het inderdaad een uitwisselmogelijkheid biedt tussen verschillende merken computers, is het voor de C-64 een uitstekende programmeerhulp. Voor andere computers met een uitgebreidere Basic wellicht een stap terug. Voor die computers blijft alleen de eerste toepassing van kracht. Er is echter zoveel programmatuur in omloop voor evenvele soorten computers, dat een uitwisseling van software tussen gebruikers van verschillende computersoorten nauwelijks zinnig lijkt. Commerciële programma's zullen niet kunnen worden gebruikt en dan is er vaak voor elke computersoort een eigen versie van het zelfde programma. Het zal dus nooit verder gaan dan het uitwisselen van zelfgemaakte software volgens het Basicode-3 protocol. Ik vraag me af op welk vlak een dergelijke uitwisseling enige zin heeft, behalve dan de vele uurtjes genoegen die de gebruikers van Basicode gegarandeerd zullen beleven. Misschien is er in het onderwijs, waar de computerchaos steeds verder doorwoekert, nog wat eer te behalen voor Basicode-3. Zoals het nu lijkt zullen de leerkrachten toch zelf aan het programmeren moeten slaan, als ze de computer werkelijk gestalte willen geven in het onderwijs. Veel educatieve uitgeverijen hebben de strijd om het marktaandeel van de educatieve software (dat er dus niet was) al opgegeven. Al die verschillende computertjes op verschillende scholen zouden wel eens nut kunnen hebben van Basicode-3. Voor de rest is Basicode echt iets voor hobbyisten, maar is daarom niet minder waardevol. We wensen Basicode-3 een lang en actief leven toe, met iets meer bekendheid dan tot nu toe het geval was.

De voordelen van programmeren in machinetaal boven programmeren in Basic zijn zo langzamerhand wel bekend. Omdat de codes niet vertaald hoeven te worden, maar direct door de microprocessor begrepen kunnen worden, loopt een machinecodeprogramma aanzienlijk sneller. Maar de snelheid is niet het enige voordeel. In machinetaal zijn ook andere zaken mogelijk die in Basic beslist niet mogelijk zijn. Hiertoe behoren onder andere de Basic-uitbreidingen.



Maak een eigen Basic-uitbreiding

Bij het ontwerpen van Commodore Basic hebben de programmeurs van Microsoft met het oog op toekomstige uitbreiding op verschillende plaatsen aan de Basic-interpreter indirecte JMP- instrukties toegevoegd.

Dat is handig, want dat zijn de aangrijpingspunten om zelf eens wat extra commando's te gaan maken. Dat gaat in machinetaal en we moeten dus weten hoe en waar we kunnen ingrijpen. De betreffende JMP's maken gebruik van RAM- vectoren. Door de waarden van die vectoren te veranderen kan men ingrijpen in de loop van Basic en de computer naar een eigen routine laten springen.

Adressen

De betreffende vectoren staan van $0300 tot $030B en we geven hieronder wat nadere informatie over wat we er mee kunnen doen.

$0300-0301 Error Vector (normaal: $E38B)

Als de computer een foutmelding wil geven, of alleen maar de melding 'ready.', wordt het x-register met een bepaalde waarde geladen en volgt een indirecte JMP ($0300). Als de waarde van x dan $80 is, wordt alleen 'ready.' geprint ($A474), bij waarden tussen 1 en 30 volgt een foutmelding ($A43A). Welke dit zijn kan men eenvoudig testen met POKE 781,nSYS 58251, waarbij n een getal is tussen 1 en 30.

Door de vector te wijzigen kan men zelf bepalen wat er moet gebeuren bij het optreden van een fout. Men kan zo bijvoorbeeld nederlandstalige foutmeldingen maken.

Het voorbeeldprogramma geeft steeds het nummer van de fout i.p.v. de volledige naam:

```
C000 LDA #$OB
C002 STA $0300 (low byte)
C005 LDA #$CO
C007 STA $0301 (high byte)
C00A RTS
C00B TXA
C00C BPL $C011
C00E JMP $A474 (ready.)
C011 PHA
C012 JSR $FFCC (sluit kanaal)
C015 LDA #$00
C017 STA $13
C019 JSR $AAD7 (regel wit)
C01C LDA #$6B
C01E LDY #$A3
C020 JSR $AB1E
     (error)
C023 LDA #$20
C025 JSR $AB47 (spatie)
C028 PLA
C029 TAX (low byte)
C02A LDA #$00 (high byte)
C02C JSR $BDCD (print getal)
C02F JMP $A46C (rest error-routine,
     o.a. in linenr.
```

$0302-0303 Warm Start Vector (normaal: $A483)

Na "ready." volgt een cursor en kan men een nieuwe opdracht ingeven, die dan uitgevoerd wordt. Klinkt logisch, maar in principe hoeft dit niet zo te gaan. Men kan namelijk door de Warm Start Vector te veranderen zelf bepalen wat dan gebeurt. Mogelijkheden zijn: een andere invoerroutine (bijvoorbeeld met funktietoetsen), allerlei trucs (zoals een AUTO-regelnummerfunktie) en direct mode commando's. Ik wijs erop, dat voor het programmeren van genoemde trucs een goede kennis van de normale routine nodig is. Het is dan ook raadzaam deze te bestuderen. Het voorbeeld toont een utility, waarbij als men niets invoert, dus alleen een return geeft, de list verschijnt:

```
C000 LDA #$0B
C002 STA $0302
C005 LDA #$C0
C007 STA $0303
C00A RTS
C00B JSR $A560 (invoerroutine)
C00E STX $7A
C010 STY $7B (init CHRGET-pointer)
C012 JSR $0073 (lees eerste char)
C015 TAX
C016 BNE $C026 (wél invoer)
C018 JSR $E544 (scherm schoon)
C01B LDA $2B
C01D STA $5F
C01F LDA $2C
C021 STA $60 (bepaal begin list)
C023 JMP $A6C3 (list-routine)
C026 JMP $A490 (normale warm start)
```

$0304-0305 Tokenize Vector normaal: $A57C)
Ingevoerde tekst wordt om geheugenruimte te sparen eerst ge'tokenize'd. Dit komt er op neer, dat elk keyword wordt vervangen door een 'token' van één byte (met een waarde van $80 of hoger). Om een lijst te maken van deze tokens het volgende Basic-programma. Getoond worden waarde in hex en decimaal, startadres van de betreffende routine en natuurlijk de naam van het keyword (run voor printer, run 110 voor scherm).

```
100 OPEN 4,4:CMD 4
110 A=41118:B=40972:T=128
120 H$="0123456789ABCDEF"
130 D=T:GOSUB 270:PRINT T;
140 P=PEEK(B)+256*PEEK(B+1)
150 IF T<163 THEN 180
160 IF T>179 AND T<203 THEN 190
170 PRINTSPC(5):B=41042:GOTO220
180 P=P+1
190 D=INT(P/256):GOSUB270
200 D=P-256*D: GOSUB 270
210 PRINT" ";:B=B+2
220 P=PEEK(A):IF P>127 THEN 240
230 PRINTCHR$(P);:A=A+1:GOTO220
240 PRINTCHR$(PAND127):A=A+1
250 T=T+1:IF T<204 THEN 130
260 CLOSE 4: END
270 H=(DAND240)/16:L=DAND15
280 PRINTMID$ (H$,H+1,1);
290 PRINTMID$(H$,L+1,1);
300 RETURN
```

De waarde van deze vector veranderen zal niet vaak nodig zijn. Alleen als er veel commando's aan Basic toegevoegd worden (zoals bij Simon's Basic) is het belangrijk om de commando's te tokenizen om tijd te winnen bij het uitvoeren van een programma.

$0306-0307 List vector (normaal: $A71A)
Als men op genoemde wijze zijn eigen tokens heeft gemaakt, moet men ook zorgen dat deze tijdens het listen weer op de juiste wijze gedecodeerd worden. Daartoe is deze List Vector aanwezig. Steeds als een karakter of keyword uit een listing op het scherm moet komen, gebeurt dat via deze vector. Men kan hem ook gebruiken voor trucjes. bekend is het 'stopzetten' van de listing door de shift-toets ingedrukt te houden. Het voorbeeld zet alle tekst uit een programma in het wit op het scherm:

```
C000 LDA #$0B
C002 STA $0306
C005 LDA #$C0
C007 STA $0307
C00A RTS
C00B BIT $0F (binnen ah-tekens?)
C00D BPL $C02A (nee)
C00F CMP #$22 (ah-teken?)
C011 BEQ $C02A (ja)
C013 STA $02 (bewaar)
C015 LDA $0286 (kleur)
C018 PHA
C019 LDA #$01 (wit)
C01B STA $0286
C01E LDA $02 (haal terug)
C020 JSR $AB47 (print char)
C023 PLA
C024 STA $0286
C027 JMP $A700 (verder listen)
C02A AND #$FF (herstel vlaggen)
C02C JMP $A71A (list als normaal)
```

$0308-0309 Character Dispatch Vector (normaal: $A7E4)
De bekendste manier om een commando aan Basic toe te voegen gebeurt met deze vector. De 'character dispatch' routine wordt telkens aangeroepen wanneer er een karakter gelezen moet worden om een commando uit te voeren. De normale routine zit zo in elkaar:
- lees een karakter uit het programma met de routine CHRGET (JSR $0073)
- is het een : neem dan het volgende karakter;
- is het het einde van de regel, ga dan naar de volgende regel;
- is het een token (≥$80) voer dan een commando uit;
- is het een letter, voer dan een let-commando uit;
- anders: geef een syntax error.

Men kan op verschillende manieren gebruik maken van de vector:
- zelf bijgemaakte tokens kunnen worden verwerkt;
- men kan commando's van meerdere letters definiëren; in nummer 5 (jaargang 3) is hierover uitvoerig gesproken;
- men kan éénkaraktercommando's maken; zo kan men bijvoorbeeld het commando introduceren;
- men kan bestaande tokens een (andere) funktie geven; zo kan men een if...then...else-instruktie toevoegen; ook is het mogelijk om commando's die normaal geen funktie hebben van een funktie te voorzien, bijvoorbeeld not heeft geen primaire funktie;
- trucjes, zoals trace, dat bij ieder commando het regelnummer op het scherm zet.

Mijn voorbeeld definiëert het POS x,y commando, waarmee men de positie van de cursor op het scherm kan regelen.

```
C000 LDA #$0B
C002 STA $0308
C005 LDA #$C0
C007 STA $0309
C00A RTS
C00B JSR $0073 (chrget)
C00E CMP #$B9 (pos?
C010 BEQ $C018
C012 JSR $0079 (nogmaals zelfde char)
C015 JMP $A7E7 (normale char disp)
C018 JSR $0073 (volgende char)
C01B JSR $B79E (parameter)
C01E STX $D3 (kolom op scherm)
C020 JSR $B7F1 (komma+parameter)
C023 STX $D6 (regel op scherm)
C025 JSR $E56C (cursorzaken)
C028 JSR $0079
C02B JSR $AAA2 (print routine)
C02E JMP $A7AE (einde char disp)
```

voorbeeld: POS 10,6;'COMMODORE"

$030A-030B Evaluate vector (normaal: $AE86)
Deze werkt ongeveer hetzelfde als de vorige. Alleen betreft het hier geen primaire commando's, maar de verwerking van numerieke of alfanumerieke parameters (secundair). Funkties als INT en CHR$ worden hier verwerkt. De aanwezigheid van deze vector biedt ongeveer dezelfde mogelijkheden als de character dispatch vector. Een voorbeeld is het toelaten van hexadecimale getallen. Hierbij wordt dan eerst getest op het $-teken, in welk geval het inlezen van het hex-getal kan beginnen.
Het wat langere voorbeeld bij deze vector maakt een nieuwe funktie aan, EVAL (string), bestaande uit de letter e en het token val. De funktie verwerkt de string tot een numerieke waarde. EVAL("3+4") geeft zo 7 als uitkomst (vgl. de VAL-funktie, waarbij de uitkomst hiervan 3 is).

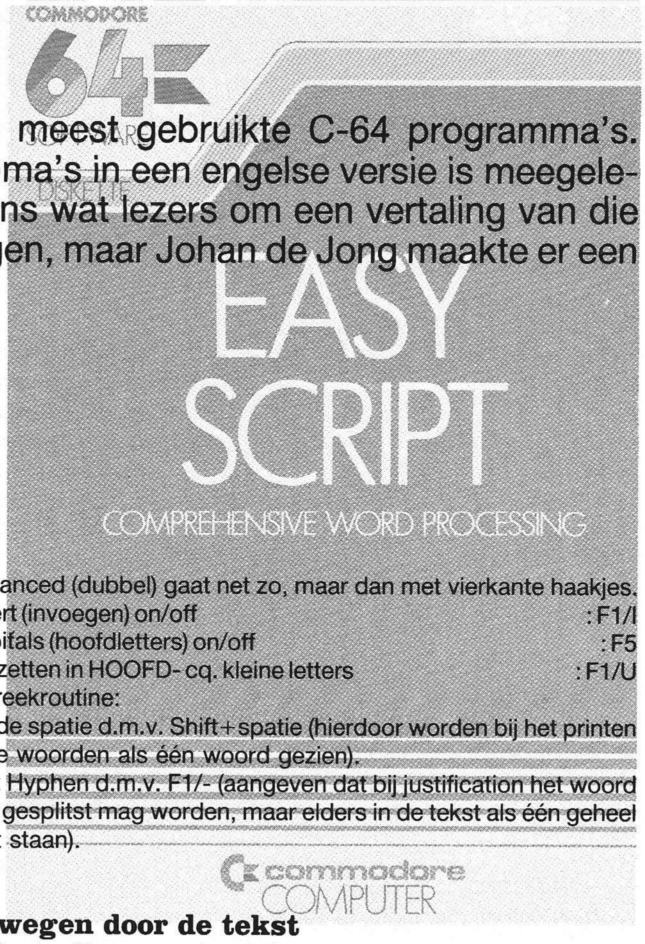
```
C000 LDA #0B
C002 STA $030A
C005 LDA #$C0
C007 STA 030B
C00A RTS
C00B LDA #$00
C00D STA $0D (numeriek)
C00F JSR $0073 (lees char)
C012 CMP #$45 (letter e)
C014 BNE $C01E
C016 LDY #$01
C018 LDA ($7A),Y (lees char erna)
C01A CMP #$C5 (val?)
C01C BEQ $C024
C01E JSR $0079 (nogmaals zelfde char)
C021 JMP $AE8D (norm. routine)
C024 JSR $0073
C027 JSR $0073 (2x chrget)
C02A JSR $AEFA (check '(')
C02D JSR $AD9E (lees expressie)
C030 JSR $AEF7 (check ')')
C033 JSR $B3A6 (direct mode mag niet)
C036 JSR $B6A3 (expr.→string)
C039 CMP #$58 (te lang?)
C03B BCC $C040
C03D JMP $a571 (string too long)
C040 TAY
C041 LDA #$00
C043 BEQ $C047
```

C045 LDA ($22),Y (byte uit string)
C047 STA $0200,Y (in buffer)
C04A DEY
C04B BPL $C045
(alle chars van de string zijn nu gekopieerd naar de input- buffer; achter de laatste byte staat een nul)
C04D LDA $7A
C04F PHA
C050 LDA $7B
C052 PHA (bewaar chrget-pointer)
C053 LDA #$00
C055 STA $7A
C057 LDA #$02
C059 STA $7B (chrget-pointer-buffer)
C05B JSR $A579 (tokenize buffer)
C05E JSR $0073
C061 JSR $AD8A (lees num. expressie)
C064 PLA
C065 STA $7B
C067 PLA
C068 STA $7A
C06A RTS

Voorbeeld van een basic-programmaatje dat gebruik maakt van de zojuist gecreëerde EVAL-funktie:

```
100 INPUT"SOMMETJE";SOM$
110 UUIT=EVAL(SOM$)
120 PRINT SOM$;"=";UIT
130 PRINT:GOTO 100
```

Zoals u in de verschillende voorbeeldprogramma's heeft gezien, zijn er nogal wat subroutines die gedienstig zijn bij het programmeren van uitbreidingen en utilities. Het is een goed idee om er een lijst van bij te houden die u in de loop van de tijd kunt uitbreiden. In ieder geval is het heel belangrijk om te begrijpen hoe bepaalde routines werken (zoals b.v. de character dispatch routine), omdat men pas dan zijn eigen routines erop kan aansluiten.
In de handel zijn verschillende boeken die kunnen helpen met het onderzoeken van Commodore-Basic. Er zijn zelfs boeken met een complete romlisting met uitleg. Maar het is natuurlijk veel leuker om het zelf allemaal uit te zoeken.
Marc de Hingh ●



De tekstverwerker Easyscript is een van de meest gebruikte C-64 programma's. Omdat echter een groot deel van die programma's in een engelse versie is meegeleverd met 1541 diskdrives, vragen er nogal eens wat lezers om een vertaling van die handleiding. Dat zou echter te veel ruimte vragen, maar Johan de Jong maakte er een korte compilatie van.

Easy script

Kort overzicht van de belangrijkste en meest gebruikte funkties en instellingen van dit populaire tekstprogramma voor de C-64.

In de tekst wordt met schuine streep (backslash) bedoeld het indrukken van de aangeduide letter na het indrukken van de funktietoets.

Bijvoorbeeld F1/L betekent: Druk F1 daarna L.

Bladspiegel

Funktietoets F3 geeft ★ (=command-mode).
Instellingen in de command-mode moeten gescheiden worden door een dubbele punt.

lmXX:rmXX =	linker- en rechtermarge.
vpXX =	marge bovenkant.
hlXX:hrXX =	marges voor header en footer.
plXX:tlXX =	paginalengte en tekstlengte.
ju =	justification (uitlijnen van de tekst).
cn =	centreren van een stuk tekst.
ra =	tekst rechts aansluitend.

Deze ju, cn en ra kunnen zijn 0 of 1 (uit of aan).
XX = aantal posities cq. regels.

De meest voorkomende instelling is:
★lm8:rm73:p166(of72):t160(of66):ju1
p166 voor 11 inch en p172 voor 12 inch-papier.

Header: kopregel

★hdXX:tekst,tekst,tekst (tekst kan dus ook blanco zijn)
↑ ↑ ↑
(left,center,right)
XX = afstand in regels tussen HEADER en TEKST.

Footer : voetregel

★ftXX:tekst,tekst,tekst
XX = afstand in regels tussen FOOTER en END OF PAGE (bv.XX=4).

Voorbeeld: Automatische paginanummering:
★ft4:,- # -,
Of voor een tekst onderaan iedere pagina:
★ft4,-# -, EASY SCRIPT
Aanduiding voor nummer (#) d.m.v. F1+shifted 3.

Saven en laden

F1/L (laden) : Vraagt om de File-naam.
F1/F (saven) : Vraagt om de File-naam.
Diskmode d.m.v. F4 (bv.Directory met $).

Te printen tekst beïnvloeden

Reversed (wit op zwart) : F1/(tekstF1/)
Enhanced (dubbel) gaat net zo, maar dan met vierkante haakjes.
Insert (invoegen) on/off : F1/I
Capitals (hoofdletters) on/off : F5
Omzetten in HOOFD- cq. kleine letters : F1/U
Afbreekroutine:
Harde spatie d.m.v. Shift+spatie (hierdoor worden bij het printen twee woorden als één woord gezien).
Soft Hyphen d.m.v. F1/- (aangeven dat bij justification het woord hier gesplitst mag worden, maar elders in de tekst als één geheel blijft staan).

Bewegen door de tekst

Dit heet editen en gaat met de cursor toetsen:

Cursor keys :	Rechts/Links/Omlaag/Omhoog
ClrHome :	Linksboven Scherm
Shift+ClrHome :	Begin tekst
←:	Einde voorgaande regel

Shift+Return : Begin volgende regel
← +Cursor→ : Begin huidige regel
Ctrl + W : Begin volgend woord
Ctrl + ← : Eind volgend woord
GOTO:
F1/G/XX : Naar regelnummer XX
F1/G/E : Naar einde tekst
F1/G/999 : Naar einde beschikbare ruimte
F1/Spatie : Volgend scherm
F1/Shift + spatie : Vorig scherm
F1/Cursor : Scrollen van de tekst
Versnellen met SHIFT
Pauze met SPATIE
Stoppen met Run/Stop

Wijzigen van tekst

Delete (teken of ruimte verdwijnt) voor :

Letter :	InstDel
Regel :	F1/InstDel

Nogmaals InstDel of weer F1
Gebied (range) : F1/D met cursor range aangeven + Return.

Erase (ook weghalen, maar ruimte wordt blank):

Letter :	Overschrijven met spatie
Regel :	F1/E/S
Paragraaf :	F1/E/P
t/m Einde tekst:	F1/E/R
Gehele tekst :	F1/E/A
Spatie tussenvoegen :	Shift+InstDel
Regel tussenvoegen :	F1/Shift+InstDel

Nogmaals Shift+InstDel of weer F1
Insert aan/uit : F1/I

Blokken tekst verplaatsen of kopiëren

Gebied (range) aangeven met F1/R/Cursor.

Verplaatsen:	Cursor op beginpositie F1/X
Kopiëren:	Cursor op beginpositie F1/A

Bekijken van tekst

(In Output-formaat) : F1/O/V

Dan door de tekst bewegen met :

Commodore toets:	regel naar beneden
Cursors horizontaal:	van links naar rechts en v.v.
F5:	20 kolommen naar rechts
F7:	40 kolommen naar rechts
Return:	Naar eerste kolom
Spatiebalk:	Snel scrollen
Spatiebalk opnieuw:	Stopt scrollen
Volgende pagina :	C = 1 regel scrollen V = volgend scherm

Stoppen met RunStop

Tabulatie

Instellen tabulatie: cursor op gewenste positie F1/T dan H of V voor horizontale c.q. verticale TAB.

Het laten zien van de Tab-posities gaat d.m.v. F1/P.

Naar Tabulatie springen met F7 c.q. F8 naar horizontale of verticale tab.

Verwijderen tabulatie: cursor op gewenste positie F1/C dan H of V om deze tab te verwijderen.

Alle tabs weg met F1/Z dan H of V (hor. of vert.).

Nieuwe pagina

★fpXX

Als er minder dan XX regels over zijn, dan wordt er op een nieuwe pagina verder geprint.

Dit wordt gebruikt als een volgend stuk tekst bij elkaar op één pagina geprint moet worden.

Als XX=0 wordt er altijd op een nieuwe pagina geprint.

Notities

★nb'notities'

Hiermee kunnen opmerkingen in de tekst vermeld worden zonder dat deze geprint worden.

Bv. boven aan een tekst:

★nb'filenaam=document'.

Printen van tekst

F1/O/P : Stopt na elke pagina. Doorgaan met C.

F1/O/CP : Print gehele tekst.

Naar OutPut Video (op beeldscherm) d.m.v. V bij een page-break i.p.v. C.

Printen stoppen met RunStop.

Geselecteerde pagina printen.

F1/O/V

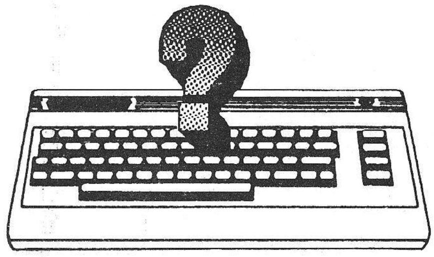
Scrollen tot aan page-break voorafgaand aan de gewenste pagina. Druk dan P.

Druk C voor de volgende pagina of V voor OutPut Video.

Voor meer functies en instellingen raadplege men natuurlijk het instructieboek. ●

Johan de Jong.

Vragen van Gebruikers.



Ook nu weer gaat Jan Bodzinga in op vragen en problemen van onze lezers.
Zijn rubriek blijkt zeer populair, maar laat dat u niet weerhouden uw vragen of opmerkingen in te sturen. Uiteraard zijn ook tips en handigheidjes van uw kant welkom.

Bulletinbord

Van vele kanten bereiken ons de verzoeken om gegevens te verstrekken over de inmiddels poulair geworden 'Bulletinbords'. Men wil graag over zoveel mogelijk telefoonnummers beschikken om, waar dan ook, een telefonisch contact te bewerkstelligen met hulp van de Commodore.

Met de regelmaat van de klok worden er nieuwe bulletinbord-services opgericht. Overal in het land zijn er hobbyisten bezig met het aansluiten van hun Commodore of andere computer aan het openbare telefoonnet, waardoor computerfans die in het bezit zijn van een **modem** met deze computers kunnen communiceren. Uit Vlaardingen ontvingen we een bericht van twee **'Sysops'**, Arjan & Michel, dat vermeldde dat daar onlangs ook een BBS is gekomen. Hun Baudrate ligt op 300/1200 en het telefoonnummer is 010-4355154 of 010-4341881. Ze zijn bereikbaar in het weekend, globaal van 18.00 tot 22.00 uur.

Commodore-printers

H.Mulder uit Annen in Drente wil graag weten hoe hij vanuit Basic met z'n MPS 801 printer ook onderkasten (kleine letters) kan printen.

De Commodore MPS 801 printer heeft net als bijna alle andere Commodore printers de eigenaardigheid, dat er standaard alleen wordt geprint in **karakterset** nummer 1. Dat betekent dat er normaal slechts hoofdletters én grafische tekens kunnen worden geprint. Om de printer naar kleine letters om te schakelen waarbij in plaats van de grafische tekens hoofdletters worden geprint, moet de printer eerst op de volgende manier worden geïnitialiseerd:

```
10 OPEN 7,4,7
20 PRINT#7
30 CLOSE 7
```

Daarna zal de printer de tekst overeenkomstig de tweede in de C-64 aanwezige karakterset op papier zetten, mits er op de bekende manier een communicatiekanaal naar de printer wordt geopend.

Audiosignalen meten op de C-64

A.Zeeuw uit IJmuiden vraagt of er niemand is die beschikt over een programma waarmee audiosignalen uit de HiFi installatie kunnen worden geanalyseerd met hulp van de Commodore 64

Het lijkt me geen eenvoudige zaak om de analoge signalen die uit de versterker komen te kunnen verwerken op frequentie en volume en dergelijke, met hulp van een eenvoudige computer. Wél zijn er hulpmiddelen op de markt, zoals de prachtige MAC 64 van Maris Electronics in Apeldoorn (plm f1.800), waarmee analoge en digitale signaaltjes kunnen worden bekeken op het scherm van de C-64 en C-128. Dit instrument zal niet zonder meer zijn toe te passen op geluidssignalen, maar kan als **oscilloscoop** z'n diensten ook in dit geval aanbieden.
Misschien is er echter iemand onder de lezers, die in de gelegenheid is meneer Zeeuw te helpen. Graag reacties.

Software-beveiliging

Regelmatig bereiken ons verzoeken of we op de redactie niet een paar trucjes weten die het kopiëren en kraken van software tegengaan. Zo ook Jeroen Verhoeve. Hij ziet z'n eigen naam maar al te vaak vervangen door die van de kraker(s).

Helaas zijn we op de redactie niet in de gelegenheid om voor iedere toepassing een beveiliging uit te denken en te verstrekken. Bovendien is het met de software voor de Commodore bijna niet meer mogelijk om alle bestaande *No-copy, Break-in, Reset en weet-ik-wat-voor-software* om andermans programma's te **kraken** en te **jatten**, het hoofd te kunnen bieden. De praktijken die vele computeraars er in deze op na houden zijn zondermeer een forse klap voor de eerlijke handel in (betaalbare) programma's. Het beste wat je kunt doen, om er zeker van te zijn, dat je eigen software in ieder geval een eerlijk Copyright-bericht blijft houden bestaat uit twee delen:
De naam van de maker via een 'SCRAMBLE-CODE' onleesbaar maken in ASCII. Dit betekent dat je er een eenvoudige rekensom op los moet laten, die de string tijdens het runnen decodeert en op het scherm zet. Je kunt bijvoorbeeld gebruik maken van een routine die alle letters ophoogt

met de positie in de string of iets dergelijks.
Deze Basic-software kan in feite alleen publiek worden verspreid als je zelf zorgt, dat er aan de listing niets meer te veranderen valt. Alle 'NO-LIST' mogelijkheden zijn inmiddels niet meer te gebruiken vanwege de vele soorten **'Final-Cartridges'** die hier weg mee weten. Het enige wat er overblijft is om je programma's met hulp van een *compiler* zodanig te vertalen, dat er in ieder geval niet met de programma's geknoeid kan worden. Op die manier bescherm je ook de codering van je copyright en eventuele andere teksten.

Meerdere printers op 8296D

De heer H.Oonk uit Hengelo gebruikt een 8296D Commodore Business computer, waarop hij een MPP 1361 printer heeft aangesloten. Hij vraagt of het mogelijk is om tegelijkertijd meerdere printers op dit systeem te kunnen aansluiten.

Om in de range van de (oude) Commodore serie business computers meerdere printers te kunnen aansluiten, kan het beste gebruik worden gemaakt van de standaard IEEE 488 bus, waarop alle **externe devices** zijn aangesloten. Het enige wat er aan de tweede en volgende printers hardwarematig zal moeten worden gewijzigd is het **devicenummer**. De IEEE-bus is zo ingericht, dat het in feite niet uitmaakt naar welk soort apparaat de paralelle data wordt gestuurd, de computer kent en *herkent* het device aan z'n nummer. Zo heeft de standaarprinter een devicenummer **4** terwijl een modem nummer **6** heeft gekregen en de floppydrive device **8**. Er kunnen maximaal zo'n 250 devices worden aangesloten, dus qua aantal kunt u nog wel even voort. In principe kunt u iedere IEEE printer gebruiken. De extra printer moet door de dealer op **device 5** worden gezet, waarna de software uiteraard de rest moet verzorgen.
Om de tweede printer los van de eerste te kunnen aansturen moeten de opdrachten worden voorzien van een OPEN-statement, waarbij device 5 wordt aangesproken. **OPEN 4,4** wordt daardoor dus **OPEN 4,5**.
U moet er wel rekening mee houden, dat bij het gebruik van commerciële software het niet in alle gevallen mogelijk is om zelf het devicenummer van de printer te kunnen aangeven. Als beide apparaten het standaar devicenummer houden, dan zullen beide printer in principe tegelijkertijd dezelfde teksten uitprinten, waarbij de de wachttijd gelijk is aan de tijd die de langzaamste printer nodig heeft om de tekst te printen!

Videobanken

G.P.Goedhart uit Zoetermeer heeft o.a. problemen met het aansturen van de Videobanken op de C-64. Hij is op zoek naar een plek in RAM, waarbij ook nog van Basic gebruik kan worden gemaakt.

De Video Interface Chip van de 64 is zo geconfigureerd, dat er een mogelijkheid bestaat om het Video-geheugen in **banken** van 16KByte op 4 verschillende plaatsen in de RAM van de C-64 neer te zetten. Het is onmogelijk voor de VIC-II om alle 64K aan RAM tegelijk te zien, maar met hulp van de zogenoemde **Bankswitching** komen we een heel eind. Om een bepaalde 16K bank als videogeheugen te kunnen bestempelen moet er worden gePOKEd. De bits die daarvoor moeten worden geconfigureerd zitten in de **6526 CIA #2** ofwel pokeadres 56578 ($DD02) en 56576 ($DD00). Poort A moet daarvoor op OUTPUT worden gezet. Dit kan gebeuren met de opdracht:

POKE 56578,PEEK(56578) OR 3

Om nu één van de vier 16K banken te kunnen kiezen moeten bit 0 en 1 van adres $DD02 worden gezet op één van de vier mogelijkheden die met het betreffende gebied in RAM corresponderen. De vier gebieden zijn als volgt ingedeeld:

%00 (0) $C000-$FFFF (49152-65535)
%01 (1) $8000-$BFFF (32768-49151)
%10 (2) $4000-$7FFF (16384-32767)
%11 (3) $0000-$3FFF (00000-16383)

De banken kunnen worden geselecteerd door de waarde van de bits (0,1,2,3) mee te geven aan het commando:

POKE56576,(PEEK(56576) AND252)OR BW

BW heeft daarbij de waarde van de bits van de betrokken BANK. Als programmeur moet er altijd rekening mee worden gehouden welke videobank actief is. De VIC-chip haalt niet alleen het schermgeheugen, maar óók de plaats van de *sprites* en *datapatronen* (bit-maps) uit dit RAM gedeelte. Als er veranderingen in de videobank worden gemaakt moet er wel in alles met deze wijzigingen rekening worden gehouden. Standaard wordt de C-64 opgestart met BW als 0, dus in de laagste geheugenbank.

80 koloms display

Keer op keer wordt de vraag gesteld of er geen software beschikbaar is, om de display van de C-64 te verbreden van 40 naar 80 karakters per regel. Gelukkig hebben we nu ook een antwoord, naar het schijnt.

We ontvingen een reactie van een lezer, die zegt een zelf geschreven oplossing voor dit probleem te bezitten. Het bestaat uit een 14 KByte groot **machinetaal-programma**, waarbij de schermroutines van 40 naar 80 tekens per regel worden gezet. Hoewel deze


SOFT WIR WAR
DOOR BERT TIER.
KIJK EENS... ECHT CHINEES AARDEWERK..
DAT ZAL HET ARTISTIEK DOEN... THUIS OP DE SCHOORSTEEN...
HOP!.. EVEN VIA MIJN VERTAAL COMPUTER VRAGEN WAT DAT MOET KOSTEN...
EUH...
BIB! BIB!
* HOEVEEL VOOR DEZE VAAS WAARDE HEER ?!..
FL 400=
WAT ZEGT 'IE ?.. HOEVEEL ?!.. HOEVEEL ?!..
EH.. FL 400=
BIB BIB
GORE AFZETTER.!!
GORE AFZETTER ?!
BIB! BIB!
GOED!!. WE GAAN DUS WEER GEWOON OVER OP HET VERTROUWDE "HANDEN EN VOETEN WERK"!
GRRR
WANT DIE VERTAAL COMPUTERTJES ZIJN LEVENSGEVAARLIJK !!!
Bert Tier '86.

software bijna de helft van de beschikbare RAM opeist, willen we dit toch aan de lezers melden. De auteur wil het programma tegen een geringe vergoeding (fl.15.-) verspreiden. Reacties op dit bericht naar: *Helmut de Bont, Klooster-ekker 12, 5541 DB REUSEL.*

Checksum

Dat niet iedereen tevreden is met de door ons gebruikte Checksum-routine blijkt uit een boze brief van W.Bos uit Veenhuizen. Hij typt nogal eens wat fouten in z'n listings en ziet dan toch een correct Checksum-getal. De waarde van ons Checksum programma ziet hij daarom niet zo zitten.

De door Commodore-Info gebruikte checksum berust op het eenvoudige en snelle principe van het tellen, per Basic-regel, van alle aanwezige ASCII waarden. Van de **som** van deze getallen wordt alleen het *LO-BYTE* bewaard en afgedrukt, om geen al te grote getallen te krijgen. Dit programma moet worden gezien als een **hulpmiddel**, en zeker niet als een *toverprogramma* waarmee alle onnauwkeurigheden als sneeuw voor de zon uit een listing kunnen worden gehaald. Daarom zal het zeker zo nu en dan gebeuren, dat er fouten in de listing worden gemaakt tijdens het intypen, die niet door de **Checksum-routine** kunnen worden achterhaald. Het devies blijft toch, dat de listings zo exact mogelijk moeten worden overgetypt. We hebben indertijd wel overwogen om een betere routine te ontwikkelen, maar dat geeft nogal wat problemen die de standaardisering van de listingverwerking in de weg staan. Jammer voor W.Bos, maar de checksum voldoet in de meeste gevallen prima.

Numeriek probleem

W.v.d.Velden uit Utrecht ontdekt dat z'n 8-bits Commodore toch niet zo nauwkeurig kan rekenen als verwacht mag worden van een computer. Met name tijdens het rekenen met diverse afrondingen vertoont het apparaat afwijkingen.

Bij het runnen van het volgende programma blijkt duidelijk wat het probleem precies veroorzaakt:

```
10 FOR I= 1 to 10 step .1
20 PRINT I,I^2
30 NEXT I
```

De getallen die nu door de computer worden geprint wijken in waarde af van ons verwachtingspatroon. De oorzaak hiervan ligt in het feit, dat de meeste *home-computers* werken met een **single precision** floating point accumulator. Dit betekent, dat alle fractionele getallen, hoe groot of klein ze ook zijn, worden opgeslagen in 8 bytes. Als gevolg hiervan kan het niet anders, of de machine moet nogal fors aan het werk met afrondingen. Het feit dat we bij bovenstaande listing te maken krijgen met getallen, die bijna de juiste waarde hebben, zoals 4.39999999 in plaats van 4.4 illustreert dit overduidelijk. In de lus van dit Basic-voorbeeld zal deze afronding weinig effect hebben; bij het werken met bedragen in een administratief programma echter des te meer! De oplossing van deze tekortkoming van de computer zullen we softwarematig aan moeten pakken. We kunnen de getallen die we door de Commodore laten berekenen en verwerken, zelf weer een juiste waarde geven, door een extra afronding te forceren. Door bijvoorbeeld de waarde die we nodig hebben te vermeerderen met een constante fractie, waardoor netjes naar boven wordt afgerond. Bij bedragen in guldens en centen helpt de volgende Basicregel:

A=INT((A+.005)*100)/100

waarbij de variabele A het te bewerken bedrag voorstelt. Op deze manier wordt de waarde van A automatisch gestript van allerlei ongerechtigheden en netjes met twee decimalen teruggezet in dezelfde veriabele. Voor getallen met meer digits achter de komma moet bovenstaande regel natuurlijk worden aangepast.

Sprite botsing

D.J.Brandt uit Rijnsburg heeft een vraag over de manier waarop botsingen tussen diverse sprites kunnen worden getest met hulp van Basic. Hij krijgt het zelf niet voor elkaar.

Hoewel er in dit blad al veel over sprites is geschreven en ook het laatste artikel over dit onderwerp nog wel niet zal zijn verschenen, toch ook in het bestek van deze rubriek een korte uitleg over dit fenomeen.
Zoals bekend kunnen er maximaal 8 sprites tegelijkertijd op het scherm van de C-64 worden getoond. Iedere sprite heeft daarbij een volgnummer van 0 tot 7, waaraan de C-64 de sprite kan herkennen. Voor ieder van deze nummers is een bit gereserveerd in het **'botsing-adres'** van de C-64 RAM. Als twee sprites elkaar raken, bereiken we, dat de bij de sprites horende **bit(s)** van 0 naar 1 worden 'gevlagd'. Deze bits blijven op 1 staan totdat er via PEEK wordt gelezen of er een botsing heeft plaats gevonden. Daarna worden de bits automatisch weer op 0 gezet. Dit **'collision-register'** vinden we op adres $D01E ofwel 53278 decimaal. De spritenummers, die lopen van 0 tot 7 kunnen alleen worden getest door het overeenkomstige bit uit dit adres te testen. Onderstaande tabel geeft de **AND-waarde** voor ieder spritenummer, waarmee de test moet worden uitgevoerd:

NUMMER	WAARDE
0	001
1	002
2	004
3	008
4	016
5	032
6	064

Om nu te kunnen testen of een sprite in botsing is geweest met één van de andere sprites, kan worden volstaan met de volgende programmaregel:

IF PEEK(53278)AND BW = BW THEN (botsing)

BW geeft hier de variabele aan waarin de bitwaarde voor de betrokken sprite volgens voorgaande tabel is opgeslagen.
Om te testen of een bepaalde sprite in botsing is gekomen met de **achtergrond** hoeven we alleen het peekadres (53278) van bovenstaande regel te veranderen in 53279! De tabel met spritenummers en bitwaarden is ook op deze test toe te passen. Denk erom, dat na het uitvoeren van een PEEK op één van deze adressen **alle** bits weer op nul worden gezet, waarna een eventuele botsing van een andere sprite, die tegelijkertijd heeft plaatsgevonden, niet meer kan worden getest. Het is dan ook een betere manier om de **complete** waarde van PEEK(53278) eerst aan een dummy-variabele toe te kennen en daarna déze variabele te testen op de hiervoor beschreven manier.

Numeriek 128 bord

J.van de Vliet vraagt of de numerieke key-pad die op het toetsenbord van de C-128 zit ook is te gebruiken in 64-stand.

Ik ben ervan overtuigd, dat er inmiddels wel een C-128 gebruiker in Nederland is, die erin is geslaagd het probleem van de heer v.d.Vliet op te lossen. Officieel is er echter geen mogelijkheid om de standaard configuratie van het C-64 bord om te zetten naar het numerieke pad op de 128. Commodore heeft dit 'bewust' op deze manier gedaan om de 128 in *alles* compatible te laten zijn met de oude C-64. Helaas blijken er elders wel enkele verschillen te zitten tussen de 64 en de 128, die we graag zouden willen ruilen voor een afwijking in compatibiliteit waar het de beide toetsenborden betreft.
Wel wil ik graag reacties van lezers, die hun oplossing voor dit probleem beschikbaar stellen. ●

In de erste aflevering van de serie "Zo zit dat...".behandelden we de "fysieke" werking van de seriebus. In dit deel gaan we ons bezighouden met het eigenlijke programmeren van de seriebus in machinetaal. Hierbij maken we gebruik van de Kernal ROM-routines in de Commodore 64, waarvan we die voor het versturen van een karakter hebben voorzien van commentaar. Als toepassing vervolgen we met een programma waarmee we de inhoud van de DOS-ROM's van de disk-drive "zeer snel" in de computer kunnen laden.

Zo Zit Dat
Het programmeren van de seriebus

door **Sjoerd Bakker**

De seriebus, ook wel IEC of Commodore bus genoemd, is de belangrijkste schakel tussen de C-64 en de randapparatuur, voor zover die tenminste daarop werkt. Dat is voor de Commodore disk-drives en printers vrijwel altijd het geval.

Bij de koppeling van de micro aan een ander apparaat via de seriebus kunnen we drie verschillende niveau's onderscheiden. Ten eerste hebben we het "bit-niveau". Dit is het niveau van het fysieke communicatieprotocol dat we in de vorige aflevering hebben behandeld. Zoals we toen zagen bestaat er een verschil tussen data-bytes, die met de ATN-lijn = hoog worden verzonden, en de opdracht-bytes die met ATN = laag worden verstuurd. In deze aflevering gaan we ons bezighouden met het versturen van deze opdracht-bytes, die bij ieder randapparaat hetzelfde effekt veroorzaken. We communiceren dan op het tweede niveau.

Bij communicatie op het derde niveau worden er tenslotte weliswaar data-bytes verstuurd, maar deze bytes worden door het randapparaat geinterpreteerd als bytes die samen een opdracht in ASCII-code vormen. In het geval van de disk-drive gebeurt dit met de data-bytes die naar kanaal 15, het opdrachtkanaal, worden verstuurd. (In het voorbeeldprogramma maken we gebruik van de M-R opdracht.) En bij (Epson-) printers worden alle data-bytes die na een escape-code, CHR$ (27), binnenkomen, behandeld als opdracht-informatie. Op die manier kun je bijvoorbeeld de onderlijning van de printer aanzetten. Opdrachten van het derde niveau hebben alleen een betekenis voor een specifiek randapparaat. Het heeft natuurlijk geen zin om je disk-drive een onderlijnopdracht te geven, of de printer een blok informatie op disk te laten zetten.

Opdrachten

Er zijn "maar" zeven verschillende opdrachten van het tweede niveau. Deze opdrachten zijn onder te verdelen in twee groepen: de eigenlijke opdracht-bytes LISTEN, TALK, UNLISTEN, UNTALK en de zgn. secundaire adressen OPEN, CLOSE en VIA (die laatste benaming is van ondergetekende). Bij het begin van de communicatie wordt eerst een eigenlijke opdracht verstuurd. Deze kan dan eventueel worden gevolgd door een tweede (Engels: second) byte: het secundair adres.

OPDRACHT-BYTES	
WAARDE	BETEKENIS
%001XXXXX	LISTEN X
%010XXXXX	TALK X
$3F	UNLISTEN
$5F	UNTALK
SECUNDAIRE ADRESSEN	
WAARDE	BETEKENIS
$FY	OPEN Y
$EY	CLOSE Y
$6Y	VIA Y
WAARBIJ X=APPARAAT-NUMMER EN Y=KANAAL-NUMMER	

Illustratie 1

Zoals uit bijgaand overzicht blijkt, dienen zowel de opdracht-bytes als de secundaire adressen te zijn voorzien van een parameter. Bits 7, 6, en 5 van de LISTEN en de TALK opdracht-byte hebben betrekking op de soort opdracht (001 = LISTEN; 010 = TALK) en de bits 4-0 geven het apparaatnummer aan (voor de seriebus: 4-31). Het versturen van de byte $24 (%001 00100) heeft dan bijvoorbeeld tot gevolg dat apparaat nummer 4, de printer, Listener wordt. Dat wil zeggen dat de informatie die door de computer daarna verstuurd wordt, uitsluitend is bestemd voor dit apparaat.

Aangezien er meerdere apparaten tegelijk aan de seriebus kunnen hangen (doorlusbaar!) voorkom je zo dat informatie die is bestemd voor de printer bij de drive terechtkomt en omgekeerd. Voor TALK gaat wat dat betreft hetzelfde op. Als je de byte $48 verstuurt geef je uitsluitend apparaat nummer 8 (de drive) de opdracht om Talker te worden, dus om informatie over de seriebus te versturen.

UNLISTEN en UNTALK hebben als code de "vaste" byte-waardes $3F

```
ED40 78          SEI
ED41 20 97 EE    JSR $EE97       DATA HI
ED44 20 A9 EE    JSR $EEA9       DATA LO ?
ED47 B0 64       BCS $EDAD       NEE: DEVICE NOT PRESENT ERROR
ED49 20 85 EE    JSR $EE85       CLOCK HI
ED4C 24 A3       BIT $A3         LAATSTE DATA-BYTE ?
ED4E 10 0A       BPL $ED5A       NEE
ED50 20 A9 EE    JSR $EEA9       DATA HI ?
ED53 90 FB       BCC $ED50       NEE
ED55 20 A9 EE    JSR $EEA9       DATA LO ?
ED58 B0 FB       BCS $ED55       NEE
ED5A 20 A9 EE    JSR $EEA9       DATA HI ?
ED5D 90 FB       BCC $ED5A       NEE
ED5F 20 8E EE    JSR $EE8E       CLOCK LO
ED62 A9 08       LDA #$08        8 BITS UITVOEREN
ED64 85 A5       STA $A5         TELLER
ED66 AD 00 DD    LDA $DD00
ED69 CD 00 DD    CMP $DD00       VASTE WAARDE ?
ED6C D0 F8       BNE $ED66       NEE
ED6E 0A          ASL             DATA HI ?
ED6F 90 3F       BCC $EDB0       NEE: TIME OUT
ED71 66 95       ROR $95         0-BIT ?
ED73 B0 05       BCS $ED7A       NEE
ED75 20 A0 EE    JSR $EEA0       DATA LO (0-BIT)
ED78 D0 03       BNE $ED7D       ONVOORWAARDELIJKE SPRONG
ED7A 20 97 EE    JSR $EE97       DATA HI (1-BIT)
ED7D 20 85 EE    JSR $EE85       CLOCK HI: BIT-WAARDE GELDIG
ED80 EA          NOP
ED81 EA          NOP
ED82 EA          NOP
ED83 EA          NOP
ED84 AD 00 DD    LDA $DD00
ED87 29 DF       AND #$DF        DATA HI
ED89 09 10       ORA #$10        CLOCK LO
ED8B 8D 00 DD    STA $DD00       ZETTEN
ED8E C6 A5       DEC $A5         8 BITS UITGEVOERD ?
ED90 D0 D4       BNE $ED66       NEE
ED92 A9 04       LDA #$04        JA
ED94 8D 07 DC    STA $DC07       TIMER LADEN
ED97 A9 19       LDA #$19        MET 1 MS.
ED99 8D 0F DC    STA $DC0F       TIMER STARTEN
ED9C AD 0D DC    LDA $DC0D
ED9F AD 0D DC    LDA $DC0D
EDA2 29 02       AND #$02        TIME OUT ?
EDA4 D0 0A       BNE $EDB0       JA
EDA6 20 A9 EE    JSR $EEA9       DATA LO ?
EDA9 B0 F4       BCS $ED9F       NEE
EDAB 58          CLI             JA
EDAC 60          RTS

.CHROUT ROM-ROUTINE SERIEBUS
```

en $5F. Omdat de computer maar één apparaat tegelijk informatie over de seriebus laat ontvangen of verzenden is het niet nodig om te specificeren welk apparaat moet ophouden met informatie te ontvangen (UNLISTEN) dan wel te verzenden (UNTALK). De computer stuurt gewoon een van deze bytes over de seriebus en vertelt zo het apparaat dat op het moment actief is zijn oren te sluiten of mond te houden.

Secundaire adressen
De secundaire adressen OPEN en CLOSE dienen om files te openen en te sluiten. Het openen van een file komt er op neer dat de file in de administratie van het betreffende apparaat wordt opgenomen. De filenaam wordt geregistreerd en er wordt een I/O-buffer toegewezen.

Door een file te sluiten wordt deze weer uit de administratie verwijderd. Van de VIA-byte zijn bits 5 en 6 wel gezet, ten teken dat het een secundair adres betreft, maar we hebben hier niet te maken met het openen of sluiten van een file. Deze byte wordt naar de drive verstuurd als we opdrachten naar kanaal 15 (het commando-kanaal) willen sturen. Bij de secundaire adressen bevat alleen de linker nybble (bits 7-4) de eigenlijke opdracht. De rechter nybble (bits 3-0) heeft betrekking op het kanaal-nummer. Het kanaal-nummer is van belang voor de toewijzing van een I/O-buffer in het randapparaat. Er zijn namelijk meestal meer kanalen (15) dan buffers in het apparaat. De opdracht $F0 aan de disk-drive heeft bijvoorbeeld tot gevolg dat een file via kanaal 0 wordt geopend, welke door de drive wordt gereserveerd voor het laden van programma's. Met $F1 open je een file via kanaal 1, dat wordt gebruikt voor het saven van programma's. En kanaal 15 (het commandokanaal) wordt dus toegewezen na het versturen van de byte $6F. Zoals gezegd beschouwt de disk-drive de data-bytes die naar kanaal 15 worden verstuurd als een specifieke disk-opdracht. Bij de behandeling van het toepassingsprogramma zullen we een voorbeeld van zo'n opdracht tegenkomen.

Seriebus-routines
Gelukkig hoeven we ons in de praktijk niet met bovenstaande, toch vrij ingewikkelde, materie bezig te houden. We kunnen gebruik maken van de algemene I/O-routines in de Kernal. Door vooraf aan te geven met welk apparaat we willen communiceren (4-31) zoeken die algemene routines automatisch de benodigde seriebus-routines op en worden deze vervolgens uitgevoerd. Toch zullen we voor de volledigheid ook de seriebus-routines even behandelen. (Zie illustratie 2) Als voorbeeld geven we de seriebus-routine voor het verzenden van een byte van de computer naar een randapparaat. Deze staat in ROM vanaf $ED40. Om het commentaar dat rechts van de listing staat te kunnen begrijpen verwijzen we naar het artikel over de seriebus in de vorige aflevering van "Zo zit dat " in Commodore-Info nr. 6 van dit jaar waarin het seriebus-protocol op bit-niveau werd behandeld. Verder is het zo, dat de testvraag voor het voldoen aan een bepaalde voorwaarde altijd zo is gesteld dat er bij een nee-antwoord wordt gesprongen. Als het antwoord ja is wordt het programma gewoon vervolgd. De sub-routines die worden aangeroepen hebben meestal betrekking op het 0 of 1 maken van een uitvoerbit in register $DD00, dan wel het naar de Carry halen van een invoerbit. De routine $ED40 verzendt dus een karakter over de seriebus naar een aangesloten apparaat. Op zich is de routine echter onvolledig. Er wordt niet bepaald of de byte met ATN = laag of hoog wordt verzonden en ook niet welke byte-waarde wordt verstuurd. De inhoud van $95, die telkens wordt geRORed voor het versturen van de 8 bits, kan natuurlijk elke denkbare waarde hebben. Vandaar dan ook dat routine $ED40 vaak door een hogere routi-

ne als subroutine wordt aangeroepen. Die hogere routine zet dan van te voren de te verzenden waarde in lokatie $95, en roept vervolgens $ED40 aan.

TALK verstuurt tenslotte de byte TALK (met apparaatnummer). Hierdoor wordt het apparaat Talker en kan het beginnen met het verzenden van informatie.

ADRES	JMP NAAR	NAAM	OMSCHRIJVING
$FF93	$EDB9	SECOND	VERSTUUR SECUNDAIR ADRES NA LISTEN
$FF96	$EDC7	TKSA	VERSTUUR SECUNDAIR ADRES NA TALK
$FFA2	$FE21	SETTMO	ZET TIME OUT
$FFA5	$EE13	ACPTR	ONTVANG KARAKTER VAN DE SERIEBUS
$FFA8	$EDDD	CIOUT	VERSTUUR KARAKTER OVER DE SERIEBUS
$FFAB	$EDEF	UNTLK	VERSTUUR UNTALK
$FFAE	$EDFE	UNLSN	VERSTUUR UNLISTEN
$FFB1	$ED0C	LISTEN	VERSTUUR LISTEN
$FFB4	$ED09	TALK	VERSTUUR TALK
OVERZICHT KERNAL SERIEBUS-ROUTINES			

Illustratie 3

Overzicht
Hierbij een overzicht van alle specifieke seriebus-routines. We zullen ze even in het kort behandelen.
SECOND verstuurt het secundair adres na LISTEN.
TKSA (TalK Secundair Adres) wordt aangeroepen om het secundair adres na TALK te verzenden. Omdat het apparaat dan Talker moet worden en de computer Listener, kan men TKSA niet aanroepen na LISTEN of SECOND na TALK. Het zijn wezenlijk van elkaar verschillende routines.
SETTMO (SET TiMe Out) wordt door de Commodore 64 niet gebruikt.
ACPTR ontvangt een byte vanaf de seriebus. De informatie-overdracht vindt plaats van het randapparaat naar de computer. Het apparaat is dan Talker en de computer Listener.
CIOUT verstuurt een byte over de seriebus. CIOUT begint niet op $ED40 omdat de bytes gebufferd worden verzonden (1 byte buffering), wat moet worden geadministreerd. Wel wordt $ED40 natuurlijk aangeroepen.
UNTLK zorgt ervoor dat de bytewaarde $5F (UNTALK) met ATN = laag over de seriebus wordt gestuurd. Het randapparaat moet vervolgens stoppen met het verzenden van informatie.
UNLSN heeft als gevolg dat de byte $3F (UNLISTEN) wordt verstuurd. Het apparaat moet hierna ophouden met luisteren.
LISTEN verstuurt de byte LISTEN (met apparaatnummer). Het apparaat moet zich gereed houden voor het ontvangen van informatie.

De praktijk
Om het nog wat ingewikkelder te maken, roepen we bovengenoemde seriebus-routines dus nooit direkt aan. Zoals gezegd gebruiken we, bij wat voor communicatie dan ook, alleen de algemene I/O-routines. Deze roepen bij seriebus-I/O op hun beurt dan weer de betreffende serie-bus-routines aan. De algemene I/O-routines hebben we behandeld in Commodore Info jaargang 3 nr.2. We zullen de door ons gebruikte routines alleen voorzien van commentaar wat betreft bijzonderheden voor de seriebuscommunicatie. (Zie illustraties 4 en 5)

```
.:CE00 80 C0 00 52 2D 4D 00 40
.
.
CE08 78          SEI
CE09 A2 07       LDX #$07
CE0B BD 00 CE    LDA $CE00,X
CE0E 95 F8       STA $F8,X
CE10 CA          DEX
CE11 10 F8       BPL $CE0B
CE13 A9 0F       LDA #$0F
CE15 A2 08       LDX #$08
CE17 A0 0F       LDY #$0F
CE19 20 BA FF    JSR $FFBA        1  SETLFS
CE1C A9 00       LDA #$00
CE1E 20 BD FF    JSR $FFBD        2  SETNAM
CE21 20 C0 FF    JSR $FFC0        3  OPEN
CE24 A2 0F       LDX #$0F
CE26 20 C9 FF    JSR $FFC9        4B CHKOUT
CE29 A2 05       LDX #$05
CE2B B5 F8       LDA $F8,X
CE2D 20 D2 FF    JSR $FFD2        5C CHROUT
CE30 CA          DEX
CE31 10 F8       BPL $CE2B
CE33 20 CC FF    JSR $FFCC        6  CLRCHN
CE36 A2 0F       LDX #$0F
CE38 20 C6 FF    JSR $FFC6        4A CHKIN
CE3B A0 00       LDY #$00
CE3D 20 CF FF    JSR $FFCF        5B CHRIN
CE40 91 FE       STA ($FE),Y
CE42 C8          INY
CE43 C0 80       CPY #$80
CE45 30 F6       BMI $CE3D
CE47 20 CC FF    JSR $FFCC        6  CLRCHN
CE4A A5 FA       LDA $FA
CE4C 49 80       EOR #$80
CE4E 85 FA       STA $FA
CE50 85 FE       STA $FE
CE52 30 D0       BMI $CE24
CE54 E6 FF       INC $FF
CE56 E6 F9       INC $F9
CE58 30 CA       BMI $CE24
CE5A 20 E7 FF    JSR $FFE7        7B CLALL
CE5D 00          BRK
CE5E 00          BRK
CE5F 00          BRK
.
.MT-LISTING 'DRIVE ROM'
```

Illustratie 4

Als voorbeeld zullen we een machinetaal-programma gaan analyseren dat de inhoud van de DOS-ROM's in de disk-drive naar de computer haalt. Hierbij maken we gebruik van de "7 stappen voor communicatie", zoals besproken in nr. 2 van dit jaar. We zullen het programma uitvoerig gaan bekijken omdat de communicatie met andere apparaten in machinetaal op praktisch dezelfde wijze geschiedt.
Het programma begint op $CE08. De lokaties $CE00-$CE07 bevat data-informatie die we nodig hebben bij de uitvoering van het programma. We beginnen met de opdracht SEI, waardoor alle IRQ-interrupts worden verboden en het programma dus wat sneller loopt. Vervolgens wordt de data-informatie van $CE00-$CE07 gekopieerd naar $(00)F8-$(00)FF. Dat is nodig omdat we gaan werken met pointers en variabele data-waardes. Vanaf $CE13 worden voorbereidingen getroffen voor de SETLFS-routine die op $CE19 wordt aangeroepen. Het file-nummer wordt 15 (A-register); het apparaatnummer wordt 8 (X-register), wat de disk-drive betekent; en het kanaalnummer in de drive wordt 15 (Y-register), het nummer van het commando-kanaal. Na SETLFS laden we het A-register met $00 en roepen we SETNAM aan. We geven hiermee te kennen dat we niet met een file-naam willen werken. Dan roepen we OPEN aan, wat vooralsnog geen merkbaar effekt heeft.

Grote lus
We komen nu terecht in de "grote lus" van het programma, die wordt ingesloten door de breedste horizontale lijnen. Hierin geven we de disk-drive telkens opdracht om $80 (128) bytes van het DOS-ROM op te zoeken en te versturen. In principe zouden we maximaal 255 bytes per keer kunnen opvragen maar helaas werkt dat, door een knullige bug in het DOS, niet bij pagina-overschrijding. De lus begint bij $CE24, waar we het X-register laden met 15 en we vervolgens CHKOUT aanroepen. Hiermee schakelen we file 15 in de computer op uitvoer naar de disk-drive. CHKOUT komt er op neer dat de LISTEN-byte (met apparaat-

nummer) en het secundair adres worden verstuurd. In dit geval worden dus achtereenvolgens de bytes $28 (LISTEN apparaat 8) en $6F (VIA kanaal 15) met ATN = laag verzonden. De drive maakt zich nu gereed om informatie via het interne kanaal 15 te ontvangen. Overigens moet men het (computer-) file-nummer 15 niet verwarren met het (drive-) kanaalnummer 15.

```
10 FOR X=52736 TO X+8*12-1
20 READ A: POKE X,A: NEXT X
30 SYS 52744: END
100 DATA 128,192,0,82,45,77,0,64
110 DATA 120,162,7,189,0,206,149,248
120 DATA 202,16,248,169,15,162,8,160
130 DATA 15,32,186,255,169,0,32,189
140 DATA 255,32,192,255,162,15,32,201
150 DATA 255,162,5,181,248,32,210,255
160 DATA 202,16,248,32,204,255,162,15
170 DATA 32,198,255,160,0,32,207,255
180 DATA 145,254,200,192,128,48,246,32
190 DATA 204,255,165,250,73,128,133,250
200 DATA 133,254,48,208,230,255,230,249
210 DATA 48,202,32,231,255,0,0,0


BASIC LOADER 'DRIVE-ROM'
```

Illustratie 5

Het is weliswaar gebruikelijk om hiervoor via SETLFS eenzelfde nummer te kiezen, maar het hoeft niet. We hadden bijvoorbeeld ook met file-nummer 1 kunnen werken. Door het aanroepen van CHKOUT zoekt deze routine van de betreffende file namelijk automatisch het bijbehorende apparaat-nummer (8) en kanaal-nummer (15) op.

M-R opdracht

Vervolgens sturen we de inhouden van de geheugenplaatsen $FD-$F8 naar de drive via de routine CHROUT. Dit zijn de data-bytes die we eerder hadden gekopieerd van $CE05-$CE00. Achtereenvolgens worden dus de waardes $4D, $2D, $52, $00, $C0, en $80 verzonden. De eerste drie bytes zijn ASCII-codes voor "M-R" (Memory-Read). Door deze opdracht naar het commando-kanaal te versturen kunnen we de inhoud van het drive-geheugen te weten komen.

De volgende twee bytes die we verzenden vormen samen het (basis)- adres waarvan we de inhoud opvragen. In ons geval is dit adres $C000. Het DOS-ROM beslaat namelijk het geheugen-gebied $C0 00-$FFFF. Van het adres $C000 versturen we eerst de Lo-byte $00 en dan de Hi-byte $C0. Dan versturen we de laatste byte uit de opdracht, $80, waarmee we aangeven dat we van $80 achtereenvolgende geheugenplaatsen de inhoud willen weten.

De totale opdracht heeft als gevolg dat de drive nu de inhouden van lokaties $C000-$C07F op gaat zoeken. CLRCHN zorgt er daarna voor dat de UNLISTEN-byte wordt verstuurd. Dan veroorzaakt CHKIN het versturen van de TALK-opdracht, waarna de drive dus Talker wordt en informatie kan gaan versturen. Via CHRIN worden de $80 bytes nu de computer binnengehaald.

De pointer op $FE/$FF geeft aan waar ze in het geheugen komen te staan. Dat wordt dus vanaf $4000. Daarna krijgen we weer CLRCHN die nu Untalk verstuurt. Vervolgens worden de pointers $F9/$FA en $FE/$FF met $80 verhoogd om hogere geheugenplaatsen in de drive te adresseren en deze ook hoger in het geheugen van de computer te zetten. Als de inhoud van $F9 "positief" wordt (bit 7 = 0) zouden we toe zijn aan lokatie $0000 in de drive, en hebben we dus alle DOS-geheugenplaatsen gehad. "Alle files" worden dan geCLOSEd. Zijn we nog niet aan het einde, dan springen we weer naar het begin van de lus en halen we de volgende $80 bytes de computer binnen.

Inefficiëntie

Schrijvende over de disk-drive, die we als voorbeeld randapparaat bij de seriebus-communicatie hebben genomen, moet me van het hart dat ik me tijdens de voorbereiding van een artikel zelden zo heb geergerd als deze keer. De inefficiëntie van het DOS is al bijna even schrikbarend als de onoverzichtelijkheid ervan. En als dat nou nog zou resulteren in een supersnelle drive... alla.

Maar dat is dus ook niet het geval. Des te schrijnender wordt zoiets als je weet dat de traagheid voor 90% is te wijten aan de communicatie-routines voor het versturen en ontvangen van "kale" byte-waardes. Door het vervangen van pakweg 100 bytes in het DOS en de computer zou de snelheid al kunnen verdubbelen, zonder dat je de specificaties voor de seriebus overschrijdt. Maar ja... de turbo-producenten varen er wel bij, zullen we maar denken. Tot zover dit artikel over het programmeren van de seriebus in machine-taal. Voor bijzonderheden wat betreft het programmeren van de drive verwijs ik naar "Het grote floppy-boek" van (het wordt eentonig) Data Becker. De mogelijkheden voor het werken met de drive vanuit Basic worden hierin echt op een voortreffelijke manier behandeld. Het boek bevat verder ook de DOS ROM-listing, voorzien van redelijk deskundig (maar vaak wel wat al te summier) commentaar. Voor het programmeren van printers zul je de handleiding bij het betreffende apparaat moeten raadplegen.

S.B. ●

Puntad '64

Voor aanvang van het schooljaar 1985/1986 schaften J.Custers, docent aan de Bisweide MAVO in Grubbenhorst, het programma Puntad '64 aan, een individuele puntenadministratie voor het onderwijs. In het kort volgen hieronder zijn ervaringen.

Het programma Puntad '64 is bedoeld voor de individuele leerkracht, die hiermee zijn eigen cijferadministratie bij kan houden.

De vele menu's in het proramma maken het mogelijk om op een eenvoudige wijze (de handleiding heb je eigenlijk niet nodig), alle mogelijke gegevens in te voeren.

Leerling- en cijfermutaties zijn eenvoudig aan te brengen en het programma biedt de mogelijkheid gemiddelden (met weegfactoren) snel te bepalen. Verder kunnen willekeurige cijfers omgerekend worden naar C-niveau, een veel voorkomende handeling voor klas 4 van de MAVO en op deze wijze erg tijdsbesparend.

Naast het feit dat ik overzichtelijke puntenlijsten kan uitprinten, kan van elke leerling een prestatiediagram in beeld gebracht worden en van elk werk een prestatieanalyse.

Het is jammer dat deze grafische weergaven niet binnen het programma kunnen worden uitgeprint. Kontakt met de makers heeft ertoe geleid dat men bezig is met een zgn. "updating", waarin deze mogelijkheid wel verwerkt wordt. Daarnaast heb ik vernomen dat het programma ook omgeschreven wordt voor "N.I.V.O.-apparatuur".

Al met al een programma waar mijn collega's en ik met volle tevredenheid mee werken.

Prijs: ƒ 125,-

Uitgebracht door: Rejasoft v.o.f. Marktstraat 7, 5991 AS Baarlo. Tel: 04707-2255 ●

Deze keer weer veel nieuws op het GEOS vlak. Deze snel populair wordende "operating shell" voor de Commodore 64 DOS heeft er de laatste tijd heel wat nieuwe applicaties en extra mogelijkheden bijgekregen. Met zaken zoals het Fontpack en Deskpack komen professionele toepassingen op de 64 steeds dichter bij.

Geos nieuws



Eindelijk kregen we de GEOS versie 1.2 in handen. Het werd tijd, Geos was (gekraakt) al bijna een eigen leven gaan leiden. Uit Amerika namen we ook nog wat andere Geos software mee, want Berkeley Softworks heeft een hele reeks Geos-utilities op stapel staan.

De grote vraag voor huidige en toekomstige GEOS gebruikers is natuurlijk: Wat is er precies veranderd? Hier volgt het resultaat van een klein onderzoek.

Andere Geos

De belangrijkste wijzigingen in het GEOS systeem liggen in het tekenprogramma Geopaint. Hiermee is het nu mogelijk geworden om tekeningen in kleur te manken. Daarbij wordt echter niet gebruik gemaakt van een multicolor HIRES scherm. De consequenties hiervan zijn dat het beeld wel een mooi oplossend vermogen behoudt, maar dat er niet met veel kleuren teglijk op een plaats gewerkt kan worden. We wezen er bovendien al op dat alleen de bezitters van een kleurenprinter voordeel van deze verandering zullen hebben. De color-mode kan echter ook uitgeschakeld worden.

Andere veranderingen t.o.v. de versie 1.0 zijn:

- de preference manager (het instelmenu) is aanzienlijk kleiner geworden en kan daardoor sneller geladen worden.

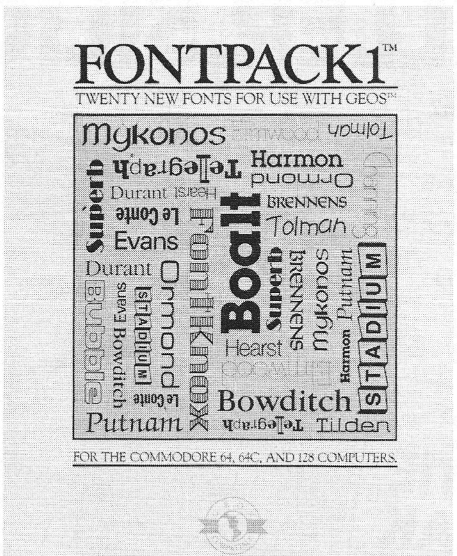


- de fouten in het printgedeelte zijn uit het programma gehaald.
- de snellaadroutine voor de diskdrive is veranderd zodat het rode lampje niet meer hinderlijk knippert.

Over het algemeen is de versie 1.2 dus aardig gedebugd, en we zijn daarom tevreden over deze veranderingen.

Extra fonts

Nieuw is ook het **Fontpack 1**. Dit bevat 20 extra fonts die in Geopaint en Geowrite



gebruikt kunnen worden. Er zitten leuke, mooie en gekke lettertypes bij, die voor veel afwisseling zorgen. Er kan overigens maar met 8 verschillende fonts tegelijk gewerkt worden. BSW, het standaardfont is altijd aanwezig. De overige zeven zijn de eerste zeven in de directory. Met Fontpack kan men in ieder geval (mits de matrixprinter daar een acceptabele afdruk van maakt) een heel eind komen in de richting van Desktop Publishing.

Deskpack I

In de Verenigde Staten al te koop, maar hier nog niet verkrijgbaar is een pakket met extra Desktop applicaties. Dit pakket bevat:

-de Graphics Grabber Met dit programma kan men graphics uit andere grafische programma's zoals The Printshop halen. De GEOS gebruiker krijgt bijvoorbeeld zo de toegang tot verzamelingen kant en klare tekeningen ('clip art') die als illustratie gebruikt kunnen worden in Geowrite of Geopaint.

-de Icon Editor, waarmee men zelf iconen (symbooltjes) kan maken.

-de Calendar (kalender), dit spreekt voor zichzelf.



-**Black Jack** (eenentwintigen tegen de computer, met muziek en graphics.)
Van deze extra utilities, die dus in de Geos omgeving functioneren en dus "muis-gevoelig" zijn, vinden we alleen de eerste twee echt interessant en nuttig. De rest neemt alleen maar plaats in beslag, spelletjes hebben de meeste C-64 gebruikers genoeg en vaak veel beter dan Black Jack.

GEOS Literatuur

Een software-systeem is net zo goed als de documentatie die erover bestaat, althans dat is de mening van Data Becker. Deze oorspronkelijk Duitse computeruitgeverij brengt daarom "**Das GroBe GEOS Buch** "uit (voorlopig alleen in het Duits). Dit boek bestaat globaal uit 3 delen :
- **GEOS fur Kreative**, met allerlei ideeen voor projecten die men met GEOS zou kunnen uitvoeren.
- **GEOS Tips & Tricks**
- **GEOS Intern** is meer voor de machinetaal programmeurs. In dit deel wordt de structuur van GEOS (in assembly) uit de doeken gedaan.

Het boek telt 350 blz. en de prijs is ongeveer f 55,-

Tips

Dan nu nog enkele tips voor het werken met de tekstverwerker **Geowrite**. Deze werkt, zoals wij al schreven, bepaald langzaam. Dit kan echter goeddeels opgevangen worden door een "misleidende" ingreep. Hiervoor moet men de kantlijnstoppen dichter bij elkaar zetten, zodat er op 2/3 van de regel niet meer gescrollt hoeft te worden.
Bovendien is het handig om de tekst eerst in BSW in te tikken en later met de muis of joystick de stijl of font te veranderen. Hiervoor moet er met de joystick en de vuurknop een bereik worden aangegeven en op dat stuk tekst (invers afgebeeld) kan men dan de verandering in font of style in het desbetreffende trekmenu inklikken.

P.B.

